1.3. Варианты заданий.

Вариант 1

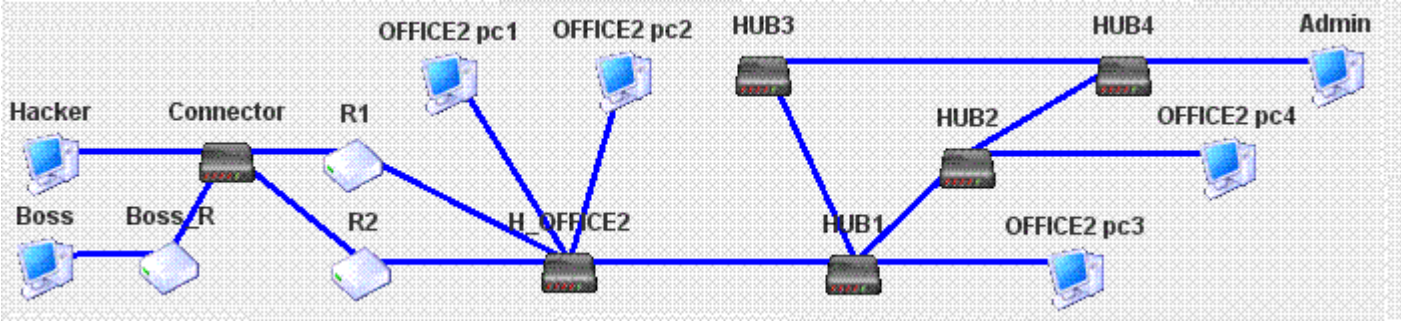

Рис. 1.2. Структура исследуемой сетевой архитектуры - вариант №1

- Файл со схемой сети: lab1_var1.jfst.
- Сеть между маршрутизаторами R1, R2 и Boss_R: 117.168.0.0.
- Компьютер Boss имеет IP-адрес 64.2.0.1.
- Компьютер Hacker имеет IP-адрес 117.168.0.5.
- Обозначения в задании: K1 – Boss, K2 – Hacker, K3 – OFFICE2 pc1.

Вариант 2

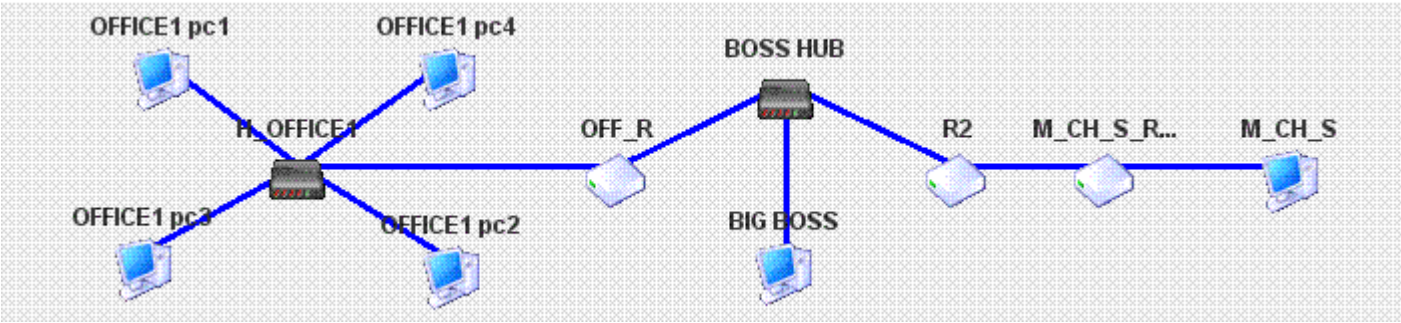

Рис. 1.3. Структура исследуемой сетевой архитектуры - вариант №2

- Файл со схемой сети: lab1_var2.jfst.
- Сеть между маршрутизаторами OFF_R и R2: 136.15.0.0.
- Компьютер BIG BOSS имеет IP-адрес 136.15.32.1.
- Компьютер M_CH_S имеет IP-адрес 10.10.0.2.
- Сеть между маршрутизаторами R2 и M_CH_S_Router: 192.178.0.0.
- Обозначения в задании: K1 – BIG BOSS, K2 – M_CH_S, K3 – OFFICE1_pc4.

Вариант 3

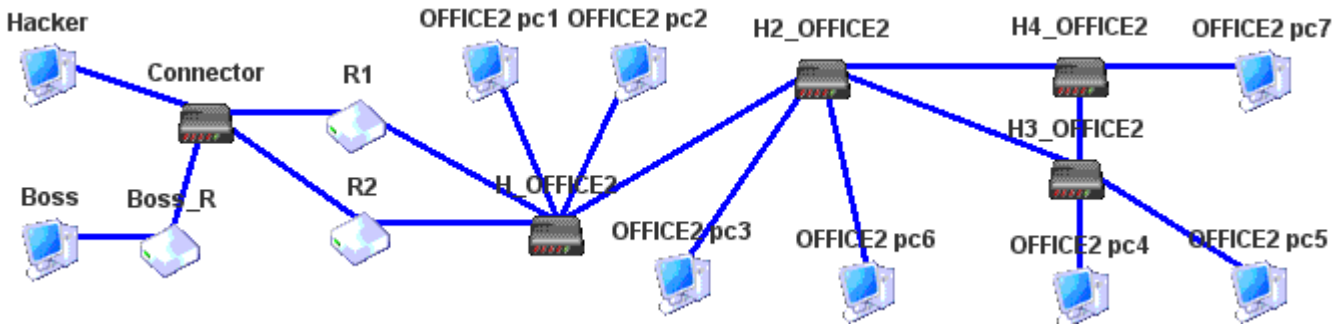

Рис. 1.4. Структура исследуемой сетевой архитектуры - вариант №3

- Файл со схемой сети: lab1_var3.jfst.
- Сеть между маршрутизаторами R1,R2 и Boss_R: 172.198.0.0.
- Компьютер Boss имеет IP-адрес 10.2.0.1.
- Компьютер Hacker имеет IP-адрес 172.198.99.252.
- Обозначения в задании: K1 – Boss, K2 – Hacker, K3 – OFFICE2_pc1.

Вариант 4

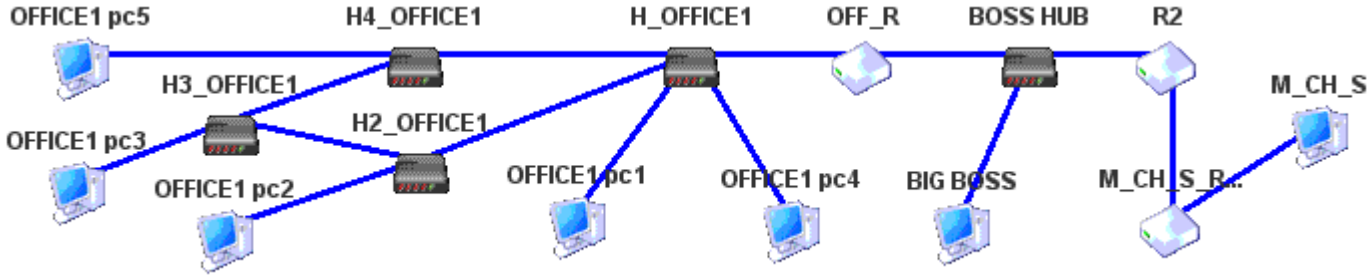

Рис. 1.5. Структура исследуемой сетевой архитектуры - вариант №4

- Файл со схемой сети: lab1_var4.jfst.
- Сеть между маршрутизаторами OFF_R и R2: 204.188.0.0.
- Компьютер BIG BOSS имеет IP-адрес 204.188.0.1.
- Компьютер M_CH_S имеет IP-адрес 10.0.0.2.
- Сеть между маршрутизаторами R2 и M_CH_S_Router: 192.178.0.0.
- Обозначения в задании: K1 – BIG BOSS, K2 – M_CH_S, K3 – OFFICE1_pc4.

Вариант 5

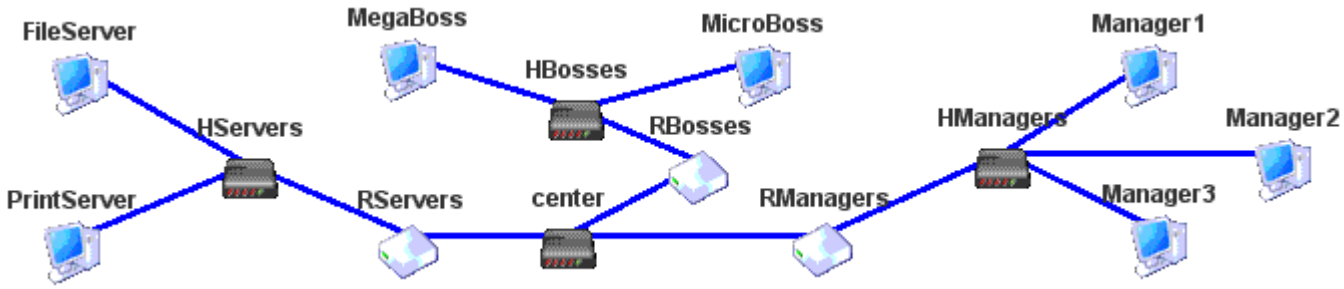

Рис. 1.6. Структура исследуемой сетевой архитектуры - вариант №5

- Файл со схемой сети: lab1_var5.jfst.
- Сеть между маршрутизаторами RServers, RManagers и RBosses: 10.0.0.0.
- Компьютер MegaBoss имеет IP-адрес 172.16.0.5.

•Компьютер Manager2 имеет IP-адрес 172.16.1.12.

•Компьютер FileServer имеет IP-адрес 172.16.10.10.

•Обозначения в задании: K1 – MegaBoss, K2 – Manager2, K3 – File-Server.

Вариант 6

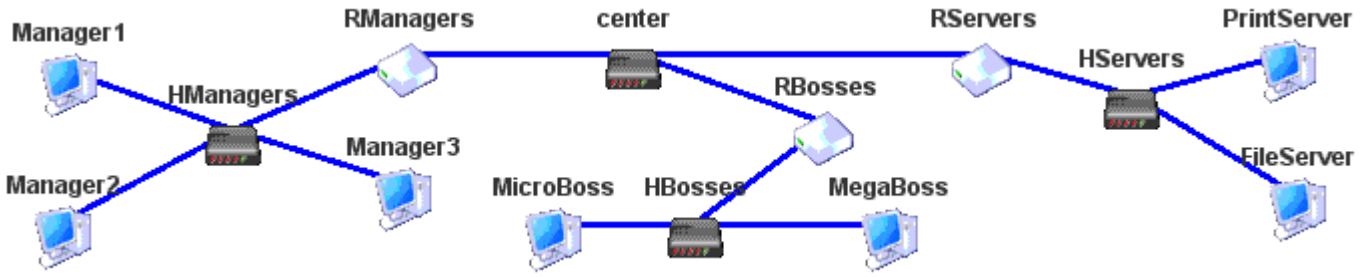

Рис. 1.7. Структура исследуемой сетевой архитектуры - вариант №6

•Файл со схемой сети: lab1_var6.jfst.

•Сеть между маршрутизаторами RServers, RManagers и RBosses: 192.168.0.0.

•Компьютер MicroBoss имеет IP-адрес 10.0.1.5.

•Компьютер Manager3 имеет IP-адрес 10.0.2.5.

•Компьютер PrintServer имеет IP-адрес 10.0.64.1.

•Обозначения в задании: K1 – Manager3, K2 – PrintServer, K3 – Micro-Boss.

Вариант 7

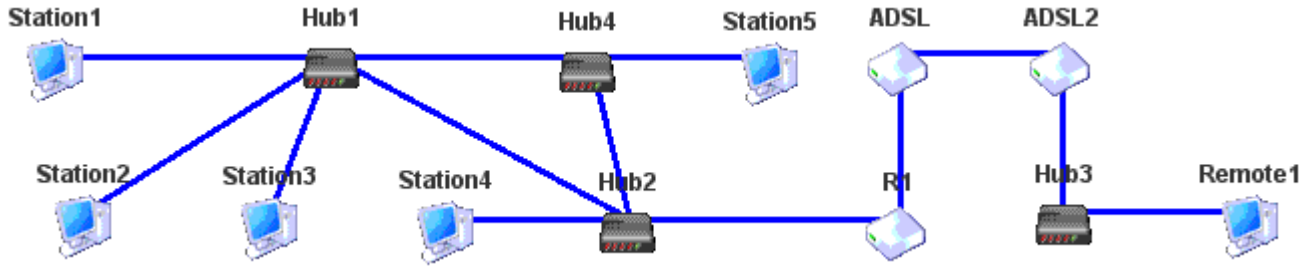

Рис. 1.8. Структура исследуемой сетевой архитектуры - вариант №7

•Файл со схемой сети: lab1_var7.jfst.

•Сеть между маршрутизаторами R1 и ADSL: 172.168.0.0.

•Компьютер Station1 имеет IP-адрес 172.168.1.2.

•Компьютер Remote1 имеет IP-адрес 10.0.0.110.

•Сеть между маршрутизаторами ADSL и ADSL2: 192.168.0.0.

•Обозначения в задании: K1 – Station1, K2 – Remote1, K3 – Station2.

Вариант 8

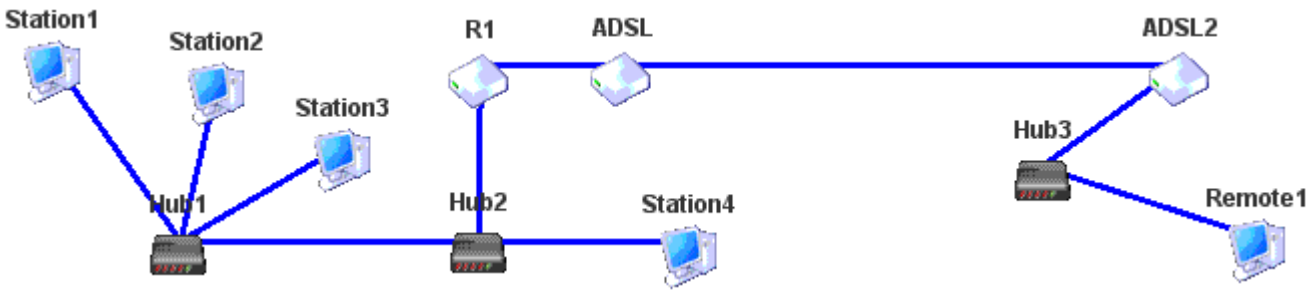

Рис. 1.9. Структура исследуемой сетевой архитектуры - вариант №8

- Файл со схемой сети: lab1_var8.jfst.
- Сеть между маршрутизаторами R1 и ADSL: 192.168.0.0.
- Компьютер Station1 имеет IP-адрес 192.168.1.2.
- Компьютер Remote1 имеет IP-адрес 99.11.0.11.
- Сеть между маршрутизаторами ADSL и ADSL2: 172.168.0.0.
- Обозначения в задании: K1 – Station1, K2 – Remote1, K3 – Station2.

Вариант 9

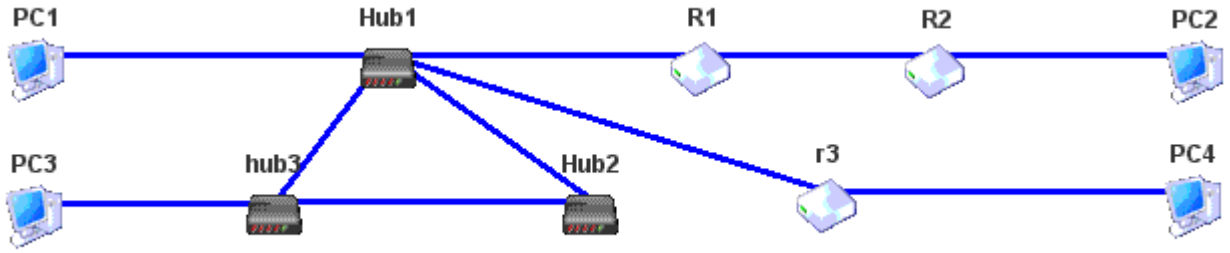

Рис. 1.10. Структура исследуемой сетевой архитектуры - вариант №9

- Файл со схемой сети: lab1_var9.jfst.
- Сеть между маршрутизаторами R1 и R2: 192.168.100.0.
- Компьютер PC1 имеет IP-адрес 129.64.128.1.
- Компьютер PC2 имеет IP-адрес 129.64.127.254.
- Компьютер PC4 имеет IP-адрес: 10.0.0.2.
- Длина маски подсети (количество значащих единиц) на PC1, PC2, PC3 должно быть минимально возможным (обеспечивая при этом корректную работу).
- Обозначения в задании: K1 – PC1, K2 – PC2, K3 – PC3.

Вариант 10

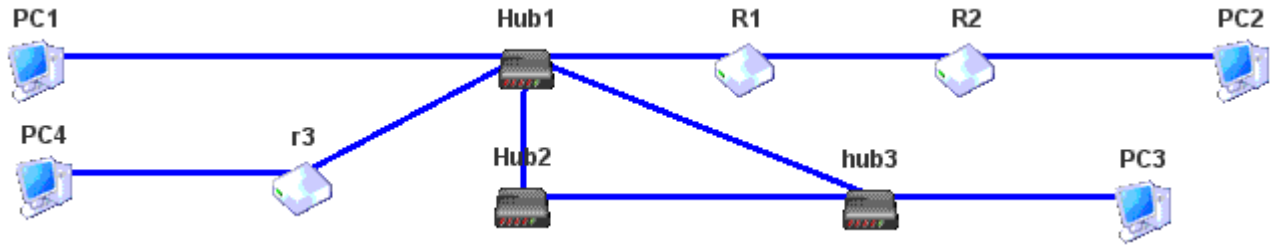

Рис. 1.11. Структура исследуемой сетевой архитектуры - вариант №10

- Файл со схемой сети: lab1_var10.jfst.
- Сеть между маршрутизаторами R1 и R2: 192.168.0.0.

•Компьютер PC1 имеет IP-адрес 172.168.0.1.

•Компьютер PC2 имеет IP-адрес 172.168.0.2.

•Компьютер PC4 имеет IP-адрес: 1.0.0.2.

•Обозначения в задании: K1 – PC1, K2 – PC2, K3 – PC3.

. Лабораторная работа №2. "Сравнительный анализ протоколов транспортного уровня TCP и UDP".

•Цель работы:

Провести анализ производительности протоколов TCP и UDP для заданной конфигурации сети, и на основании полученных результатов сделать заключение о том, какой протокол предпочтительнее использовать.

•Порядок выполнения работы:

1.В качестве схемы сети взять результат выполнения соответствующего варианта лабораторной работы №1. Установить коэффициенты прохождения пакетов согласно вашему варианту.

2.Протестировать отправку по UDP и по TCP 20 сообщений с K1 на K3.

3.Объяснить, анализируя вывод программы, какой протокол выгоднее использовать с точки зрения скорости доставки информации.

4.Протестировать отправку по UDP и по TCP 20 сообщений с K2 на K1.

5.Объяснить, анализируя вывод программы, какой протокол выгоднее использовать с точки зрения надежности доставки информации.

6.Подсчитать процент потерь пакетов. С учетом того, что должно теряться не более 7% пакетов. Объяснить, как привести сеть к требуемому лимиту по потерям.

7.Проанализировать время соединения, сделать вывод о том, какой протокол быстрее справился с поставленной задачей (необходимо учитывать требуемую надежность).

8.Определить состояние, при котором сеть начинает удовлетворять требованиям по потери пакетов. То есть подобрать такие значения коэффициентов пропускания, при которых будет теряться не более 7% пакетов.

Разрешается использовать диапазон значений длины 10, то есть можно найти интервал значений коэффициентов пропускания длины 10, где на нижней границе сеть не удовлетворяет критерию потерь пакетов, а на верхней заданный критерий удовлетворяется.

•Отчет должен содержать:

Анализ производительности протоколов TCP и UDP для заданной конфигурации сети при коэффициенте пропускания равном 100, расчет процента потерь пакетов и анализ производительности сети для обоих протоколов в условиях недоброкачественных линий передач для обоих протоколов, оценку удовлетворения сетью критерия по потере пакетов, анализ времени соединения. В отчете также необходимо привести вывод о том, какой протокол предпочтительнее использовать в данной конфигурации сети.

3.1. Пример выполнения лабораторной работы.

•Исходные данные:

•Файл со схемой сети: lab4_sample.jfst.

•Заданная конфигурация сети имеет вид, приведенный на рис.1.1.

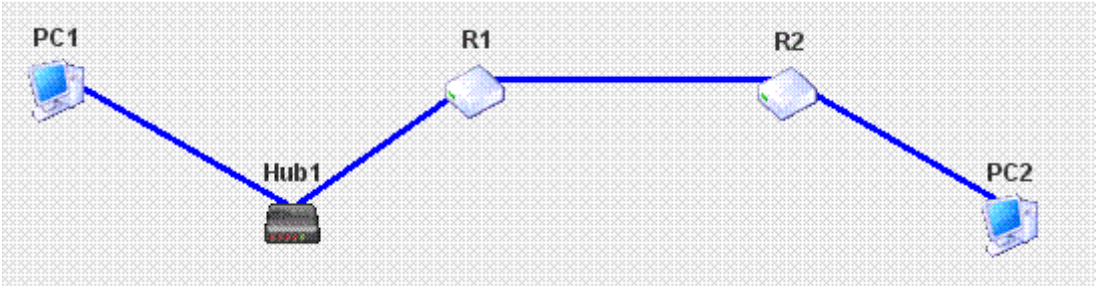

Рис. 3.1. Структура исследуемой сетевой архитектуры.

•???? Сеть между маршрутизаторами R1 и R2: 172.168.100.0.

- Компьютер PC1 имеет IP-адрес 192.168.0.1.
- Компьютер PC2 имеет IP-адрес 192.168.0.2.

- Задание:
  - Установить коэффициент пропускания всех линий равный 100.
  - Протестировать отправку по UDP и TCP 20 сообщений с PC2 на PC1.
  - Объяснить, анализируя вывод программы, какой протокол выгоднее использовать с точки зрения скорости доставки информации.
  - Установить коэффициент пропускания 65.
  - Протестировать отправку по UDP и TCP 20 сообщений с PC2 на PC1.
  - Объяснить, анализируя вывод программы, какой протокол выгоднее использовать с точки зрения надежности доставки информации.
  - Подсчитать процент потерь пакетов. С учетом того, что должно теряться не более 7% пакетов. Объяснить, как привести сеть к требуемой надежности.
  - Проанализировать время соединения, сделать вывод о том, какой протокол быстрее справился с поставленной задачей, учитывая требуемую надежность.
  - Определить состояние, при котором сеть начинает удовлетворять требованиям по потери пакетов.

Порядок выполнения будет следующим:

1.Убедимся, что коэффициент пропускания на всех линиях, в том числе между PC1 и PC2 равен 100.

2.Выберем PC1 и запустим на нем UDP-приложение (UPD-сервер), выбрав в качестве прослушиваемого порт 7. Программа выдаст следующее сообщение:

pc1 Application is now listening on port 7.

3.Выберем PC2 и пошлем через UDP-приложение 20 сообщений со строкой "rsh" на PC1. Программа выдаст похожее на следующее сообщение (для первого из двадцати сообщений):

pc2 Start sending echo message 'rsh' to 192.168.0.1:7

pc2 Created UDP packet for 192.168.0.1:7.

pc1 Created ARP Response packet to 192.168.0.2

pc1 Sending packet from ProtocolStack (to 192.168.0.2).

pc2 Sending packet from ProtocolStack (to 192.168.0.1).

pc1 ProtocolStack received packet from local Interface.

pc1 Confirmed Packet is for this Network Layer Device.

pc1 UDP packet received from 192.168.0.2:3000 message:

"rsh". UDP Port 7 has status "busy" from now.

pc1 Application Recieving echo message 'rsh' from client.

pc1 Application Sending echo message 'rsh' to client.

pc1 Created UDP packet for 192.168.0.2:3000.

pc1 Sending packet from ProtocolStack (to 192.168.0.2).

pc2 ProtocolStack received packet from local Interface.

pc2 Confirmed Packet is for this Network Layer Device.

pc2 UDP packet received from 192.168.0.1:7 message: "rsh".

pc2 Recieving echo message 'rsh' from server.

pc1 Server closing connection. Now listening on 7.

pc1 Application is now listening on port 7.

4.Выберем меню статистики узла PC1 и проверим, сколько UDP дейтаграмм он получил и отправил. Будет выведен следующий результат:

"Received UDP segments: 20" ,

что означает, что получено 20 UDP дейтаграмм, и

"Sent UDP segments: 20",

что означает, что отправлено 20 UDP дейтаграмм. При заданных параметрах сети процент потерь равен 0%, что удовлетворяет требованиям.

5.Обнулим статистику узла PC1. Теперь установим коэффициент пропускания линии между двумя узлами в значение, равное 65 и снова пошлем с PC2 на PC1 20 UDP дейтаграмм.

6.Выберем меню статистики узла PC1 и проверим, сколько UDP дейтаграмм он получил и отправил. Будет, с большой вероятностью, выведен следующий результат:

"Received UDP segments: 13"

"Sent UDP segments: 13"

что означает, что получено 13 UDP дейтаграмм, и отправлено 13 UDP дейтаграмм.

7.Выберем меню статистики узла PC2 и проверим, сколько UDP дейтаграмм он получил и отправил за все время нашего опыта. Будет, с большой вероятностью, выведен следующий результат:

"Received UDP segments: 26"

"Sent UDP segments: 40"

что означает, что получено 26 UDP сегментов, и отправлено 40 UDP сегментов. При заданных параметрах сети процент потерь больше, чем 7%, что не удовлетворяет требованиям. Можно попробовать использовать протокол TCP.

8.Выберем PC1 и запустим на нем TCP-приложение (TCP-сервер), выбрав в качестве прослушиваемого порт 8. Программа выдаст следующее сообщение:

pc1 Application is now listening on port 8.

9.Выберем PC2 и пошлем через TCP-приложение 20 сообщений со строкой "ppp" на PC1. Программа выдаст похожее на следующее сообщение (для первого из двадцати сообщений, дошедших до PC1):

pc2 Connecting to host 192.168.0.1:8.

Please wait...

pc2 TCP SYN-packet for 192.168.0.1:8.

pc1 ProtocolStack received packet from local Interface.

pc1 Created ARP Response packet to 192.168.0.2

pc1 Sending packet from ProtocolStack (to 192.168.0.2).

pc2 ProtocolStack received packet from local Interface.

pc2 Sending packet from ProtocolStack (to 192.168.0.1).

pc1 ProtocolStack received packet from local Interface.

pc1 TCP SYN-packet received from 192.168.0.2:3000.

TCP Port 8 has status "busy" from now.

pc1 Created TCP SYN-packet for 192.168.0.2:3000.

pc1 Sending packet from ProtocolStack (to 192.168.0.2).

pc2 TCP SYN-packet with ACK received from 192.168.0.1:8.

TCP Port 3000 still has status "busy".

pc2 Created TCP acknowledgement packet for 192.168.0.1:8.

pc2 Sending packet from ProtocolStack (to 192.168.0.1).

pc1 TCP packet with establishing connection ACK received

from 192.168.0.2:3000. Connection confirmed!

New TCP connection established!

pc2 Start sending echo message 'ppp' to 192.168.0.1:8

pc2 Created TCP data packet for 192.168.0.1:8.

pc2 Sending packet from ProtocolStack (to 192.168.0.1).

pc1 ProtocolStack received packet from local Interface.

pc1 Created TCP acknowledgement packet

for 192.168.0.2:3000.

pc1 Sending packet from ProtocolStack

(to 192.168.0.2).

pc2 ProtocolStack received packet from local Interface.

pc2 TCP packet with establishing connection ACK

received from 192.168.0.1:8. Connection confirmed!

pc1 TCP packet with data received from 192.168.0.2:3000.

Passing data to application program.

pc1 Recieving echo message 'ppp' from client.

pc1 Sending echo message 'ppp' to client.

10.Выберем меню статистики узла PC1 и проверим, сколько TCP сегментов он получил и отправил. Будет выведен следующий результат:

"Received TCP segments: 45",

"Sent TCP segments: 43",

"Sent TCP ACKs: 23",

что означает, что отправлено 23 подтверждения, также будет нулевая статистика по отосланным и принятым дубликатам.

Выберем меню статистики узла PC2 и проверим, сколько TCP сегментов он получил и отправил. Будет выведен следующий результат:

"Received TCP segments: 43",

"Sent TCP segments: 45",

"Sent TCP ACKs: 23",

что означает, что отправлено 23 подтверждения, также будет нулевая статистика по отосланным и принятым дубликатам.

Из этого можно сделать вывод о том, что для хорошей линии передач излишне проводить загрузку канала подтверждениями о получении сегментов, которые занимают около 50% сегментов, задействованных в обмене информацией, однако, были доставлены все 20 сообщений, что удовлетворяет требованиям по процентам потерь.

11.Обнулим статистику узла PC1 и PC2. Теперь установим коэффициент пропускания 60 и снова пошлем с PC2 на PC1 20 TCP сегментов.

12.Выберем PC2 и пошлем через TCP-приложение 20 сообщений со строкой "ppp" на PC1. Программа, в нашем случае, выдаст следующее сообщение:

pc2 Packet lost due to physical link problems!

pc1 Server awaiting connection timeout!

Now server is listening to port: 8.

pc2 Connection timeout! Closing connection to host: 192.168.0.1:8.

Это говорит о том, что на PC2 было закрыто подключение к PC1 и на PC1 было закрыто подключение к PC2, т.к. качество линий в данном примере не позволяет обмениваться информацией за установленные программой на соединение промежутки времени. При таких параметрах сеть не удовлетворяет требуемым условиям по потерям: не более 7%.

Если проверить статистику PC2, то можно увидеть, что было отправлено 14 дубликатов, а получено 27 дубликатов. В то время, как было отправлено 10 сегментов, а получено 11.

13.Обнулим статистику узла PC1 и PC2. Теперь установим коэффициент пропускания 88 и снова пошлем с PC2 на PC1 5 TCP сегментов.

14.Выберем меню статистики узла PC2 и проверим, сколько TCP сегментов он получил и отправил за время нашего опыта. Будет, с большой вероятностью, выведен результат такой, что было отправлено 14 TCP сегментов, 8 дубликатов, 6 подтверждений, также было получено 10 сегментов.

15.В результате анализа полученных результатов можно сделать следующие выводы. В условиях качественного обеспечения передачи UDP протокол показал себя с хорошей стороны, так как все дейтаграммы дошли до адресатов. По времени было затрачено 32ms. Не тратилось время на установление соединения и на подтверждения получения пакетов.

При плохом качестве линий не все пакеты дошли до пунктов назначения. Оправданием использования UDP на плохих линиях может стать только то, что информация за время задержки или потери станет неактуальна, и ее можно не передавать. К примеру, видеоконференция через Интернет.

Результаты проведенной работы по протоколу TCP говорят о неэффективном использовании данным протоколом качественных линий, так как дополнительное время тратится на подтверждение пакетов, а также на установление и разрыв связи. В условиях некачественной физической линии использование TCP явно предпочтительнее, так как "потерявшиеся" сегменты пересылаются и, в конечном счете, доходят до адресата.

По времени передача по протоколу TCP заняла 344ms, что в 10.75 раза больше, чем время затраченное при передаче через UDP. Таким образом, применение протокола оправдано в случаях, требующих гарантированного получения адресатом всей посылаемой информации. К примеру, проверка электронной цифровой подписи.

Очевидно, что при использовании UDP сеть начинает удовлетворять семипроцентному критерию по потере пакетов при коэффициенте пропускания между узлами PC1 и PC2 не менее 93%. Если использовать TCP, то критерий по потере пакетов удовлетворяется при коэффициенте пропускания между узлами PC1 и PC2, принадлежащем интервалу от 60 до 65.

3.2. Контрольные вопросы.

1. Какой из протоколов транспортного уровня обеспечивает надежную доставку данных? За счет какого механизма обеспечивается гарантия доставки?

2. Назовите ситуации, в которых применение протокола UDP является целесообразным.

3. Назовите ситуации, в которых применение протокола TCP является целесообразным.

4. Чем в TCP обеспечивается ускорение работы по передаче данных?

5. Сколькими пакетами обмениваются во время UDP-соединения клиент и сервер?

6. Сколькими пакетами обмениваются во время TCP-соединения клиент и сервер? Какие существуют пакеты TCP?

3.3. Варианты заданий.

Вариант 1

- Установите коэффициент прохождения пакетов между узлами Connector и Boss_R в 82.
- Обозначения в задании: K1 – Boss, K2 – Hacker, K3 – OFFICE2 pc1.

Вариант 2

- Установите коэффициент прохождения пакетов между узлами H_OFFICE1 и OFF_R в 71.
- Обозначения в задании: K1 – BIG BOSS, K2 – M_CH_S, K3 – OFFICE1 pc4.

Вариант 3

- Установите коэффициент прохождения пакетов между узлами Connector и Hacker в 75.
- Обозначения в задании: K1 – Boss, K2 – Hacker, K3 – OFFICE2 pc1.

Вариант 4

- Установите коэффициент прохождения пакетов между узлами BOSS HUB и R2 в 85.
- Обозначения в задании: K1 – BIG BOSS, K2 – M_CH_S, K3 – OFFICE1 pc4.

Вариант 5

- Установите коэффициент прохождения пакетов между узлами HBosses и center в 80.
- Обозначения в задании: K1 – MegaBoss, K2 – Manager2, K3 – FileServer.

Вариант 6

- Установите коэффициент прохождения пакетов между узлами HManagers и center в 78.
- Обозначения в задании: K1 – Manager3, K2 – PrintServer, K3 – MicroBoss.

Вариант 7

- Установите коэффициент прохождения пакетов между узлами Hub2 и R1 в 85.
- Обозначения в задании: K1 – Station1, K2 – Remote1, K3 – Station2.

Вариант 8

- Установите коэффициент прохождения пакетов между узлами Hub2 и R1 в 55.
- Обозначения в задании: K1 – Station1, K2 – Remote1, K3 – Station2.

Вариант 9

- Установите коэффициент прохождения пакетов между узлами Hub1 и R1 в 75.
- Обозначения в задании: K1 – PC1, K2 – PC2, K3 – PC3.

Вариант 10

- Установите коэффициент прохождения пакетов между узлами Hub1 и R1 в 65.
- Обозначения в задании: K1 – PC1, K2 – PC2, K3 – PC3.

Назначение и приемы работы в имитаторе javaNetSim

Основной задачей имитатора javaNetSim является имитация работы всех уровней стека протоколов TCP/IP. Для этого имитируется работа протоколов каждого из уровней, чем достигается полная имитация работы сети. В связи с этим имитатор javaNetSim удобен для выполнения лабораторных работ. Основные приемы работы с имитатором javaNetSim будут рассмотрены в данной главе.

Имитатор javaNetSim является объектно-ориентированным и написан на языке Java. Программы написанные на этом языке являются машинно- независимыми, т.е. имитатор javaNetSim будет работать на любом компьютере, для которого есть виртуальная Java машина. Хотя язык Java является интерпретируемым, это не оказывает существенного влияния на быстродействие имитатора. Это объясняется тем, что имитатор разрабатывался для моделирования работы небольших сетей, обработка моделей которых не требует больших вычислительных ресурсов.

Архитектура имитатора javaNetSim выглядит следующим образом. В основе лежит класс Simulation (Имитация), который содержит объекты классов Link (Линия) и Node (Узел). Этот класс предназначен для объединения устройств и линий связи в единую сеть. Класс Link содержит ссылки на объекты класса Node, и предназначен для соединения двух узлов между собой. Класс Node содержит ссылки на объекты класса Link и является наиболее общей моделью сетевого устройства.

Все реальные сетевые устройства являются производными от объекта класса Node и соответствуют модели стека протоколов TCP/IP:

- Hub (Концентратор) – DataLink Layer Device (Устройство физического уровня) – имеет пять портов, т.е. к нему возможно подключить до пяти линий связи;
- Router (Маршрутизатор) – Network Layer Device (Устройство сетевого уровня) – имеет два порта, а также стек протоколов TCP/IP (ProtocolStack);
- PC (Компьютер) – Applications Layer Device (Устройство уровня приложений) – имеет один порт, стек протоколов TCP/IP, а также возможность выполнять клиентскую или серверную часть какого-либо приложения.

Для взаимодействия с пользователем каждому сетевому устройству нужно графическое соответствие. Его обеспечивают следующие классы:

- GuiHub (Графический пользовательский интерфейс концентратора);
- GuiRouter (Графический пользовательский интерфейс маршрутизатора);
- GuiPC (Графический пользовательский интерфейс компьютера).

Как сами сетевые устройства, так и графический пользовательский интерфейс сетевых устройств должен быть единым. Этим объединением занимается класс SandBox (Рабочая область).

(начало)

5.1. Графический интерфейс имитатора javaNetSim

Рабочая область является частью основного окна программы, представленного на рисунке 5.1.

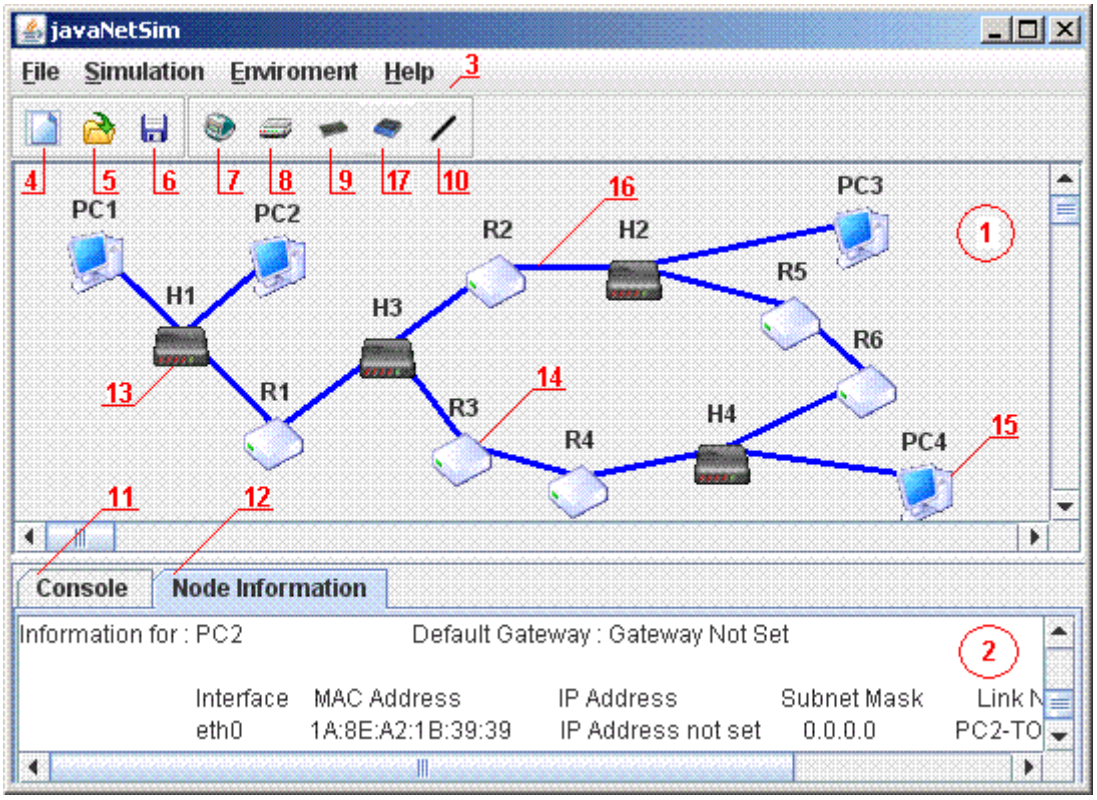

Рис. 5.1. Основное окно программы javaNetSim.

Основное окно программы логически разделено на четыре части:

1.рабочая область, обозначенная на рисунке цифрой (1) – содержит сетевые устройства и линии связи между ними:

- •Концентратор на пять сетевых интерфейсов (13).
- •Маршрутизатор соединяющий две подсети (14).
- •Компьютер или конечный узел сети (15).
- •Линия связи между двумя сетевыми устройствами (16).

2.область вывода результатов (2) – содержит две вкладки:

- •вкладка "консоль" (11) – содержит журнал передачи пакетов по сети
- •вкладка "информация об устройствах" (12) – для каждого интерфейса всех сетевых устройств содержит IP-адрес, маску подсети и шлюз по умолчанию.

3.главное меню (3) – содержит основные действия по управлению имитатором;

4.линейка инструментов – содержит следующие кнопки:

- •кнопка "создать пустую конфигурацию" (4);
- •кнопка "открыть существующую конфигурацию" (5);
- •кнопка "сохранить текущую конфигурацию" (6);
- •кнопка "создать компьютер" (7);
- •кнопка "создать маршрутизатор" (8);
- •кнопка "создать концентратор" (9);
- •кнопка "создать коммутатор" (17);

•кнопка "создать соединение" (10).

Основное окно программы представляет собой инструмент взаимодействия пользователя с имитатором. С помощью этого инструмента пользователь может добавлять, удалять и соединять между собой сетевые устройства, а также работать с сетью на любом из четырех уровней стека протоколов TCP/IP.

(начало)

5.2. Главное меню программы

Меню File(файл) позволяет создавать, открывать и сохранять конфигурации сетей для их дальнейшего использования. Меню содержит пять пунктов:

•New(Новый) – создать пустую конфигурацию.

•Open...(Открыть...) – открыть существующую конфигурацию.

•Save...(Сохранить...) – сохранить текущую конфигурацию.

•Save As...(Сохранить Как...) – сохранить текущую конфигурацию под новым именем.

•Exit(Выход) – выйти из имитатора javaNetSim.

Режим проектирования сети доступен из меню Simulation(Имитация). Это меню позволяет создавать новые сетевые устройства (такие как: концентратор, маршрутизатор или компьютер) и изменять сетевые параметры уже существующих устройств. Меню содержит два пункта:

•подменю Add(Добавить) – позволяет создать компьютер(PC), маршрутизатор(Router) или концентратор(Hub);

•подменю Tools(Инструменты), в котором есть пункт Set TCP/IP Properties(Установить свойства TCP/IP) позволяющий изменить свойства TCP/IP.

В имитаторе javaNetSim задание IP-адреса узла, маски подсети и шлюза по умолчанию происходит через диалог "Internet Protocol (TCP/IP) Properties", вызов которого осуществляется через меню "Simulation -> Tools -> Set TCP/IP Properties".

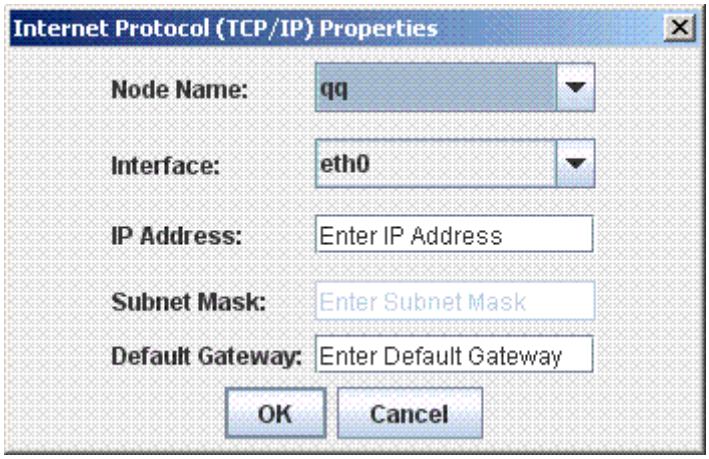

Рис. 5.2. Установка параметров TCP/IP.

В этом окне (рис. 6.2) для выбранного устройства (Node Name) и интерфейса (Interface) можно задать IP-адрес (IP Address) и маску подсети (Subnet Mask) для интерфейса и шлюз по-умолчанию (Default Gateway) для узла. Для компьютера доступен всего один интерфейс, для маршрутизатора – два.

Управление параметрами имитатора доступно из меню Environment(Окружение) и позволяет изменять режим отображения информации, а также очищать область вывода результатов. Меню содержит четыре пункта:

•Clear Console(Очистить консоль) – удаляет все записи из вкладки "консоль";

•Clear Node Information(Очистить информацию об устройствах) – удаляет все записи из вкладки "информация об устройствах";

•Show simulation messages for:(Показывать сообщения имитатора для:) – позволяет задать режим вывода на вкладку "консоль" сообщений только определенных уровней стека протоколов TCP/IP.

Есть возможность выбрать следующие уровни: Link and DataLink Layers(Физический и канальный уровни), Network Layer(Сетевой уровень), Transport Layer(Транспортный уровень), Application Layer(Уровень приложений);

•Show headers:(Показывать заголовки) – позволяет задать режим вывода на вкладку "консоль" сообщений с названиями уровней и/или с типами пакетов.

С помощью меню "Environment -> Show simulation messages for:" можно отключить сообщение от тех уровней стека протоколов TCP/IP в которых нет необходимости. Это уменьшит количество информации выводимой в "консоль" и облегчит поиск нужных данных.

(начало)

5.3. Контекстное меню

Контекстное меню, вызываемое щелчком правой кнопкой мыши, отличается для устройств работающих на разных уровнях стека протоколов TCP/IP. На рис. 6.3 изображены контекстные меню соответственно для устройств: физического уровня - концентратор (а), сетевого уровня - коммутатор (б) и прикладного уровня - компьютер (в).

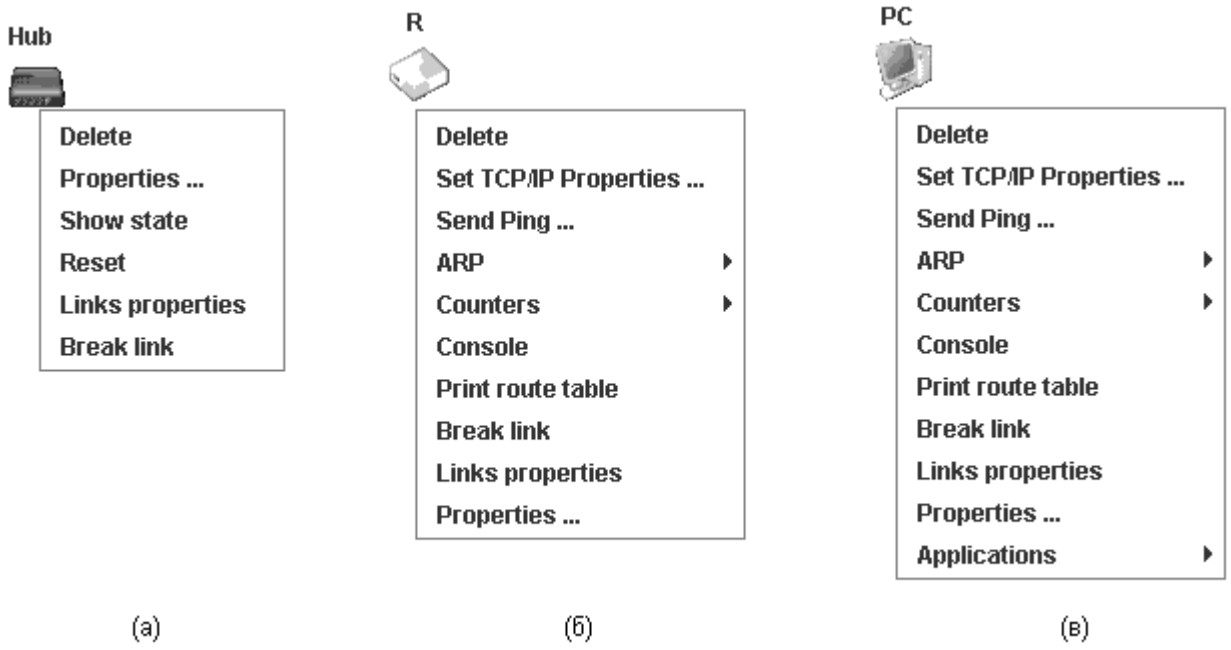

Рис. 5.3. Контекстное меню.

Основные пункты контекстного меню, общие для всех устройств перечислены ниже.

- Delete(Удалить) – без подтверждения удаляет выбранное сетевое устройство из текущей конфигурации.
- Properties(Свойства) – вызывает диалог, показывающий сетевые настройки выбранного устройства. Для каждого интерфейса показывается MAC адрес, IP адрес, маска подсети, название подключенной линии связи. Также для устройства указаны имя и шлюз по умолчанию.
- Break Link(Разорвать линию связи) – вызывает диалог, в котором можно выбрать интерфейс, линию связи которого требуется разорвать.
- Links Properties(Свойства линий связи) – позволяет установить свойства линии связи.

При выборе пункта Link Properties(Свойства линий связи) вызывается диалог, который позволяет установить коэффициент пропускания для интерфейса, показывающий какой процент пакетов линия связи подключенная к этому интерфейсу будет пропускать. Коэффициент пропускания задается для интерфейса (eth0, eth1 и т.д.).

В меню концентратора имеются два дополнительных пункта, позволяющих следить за его состоянием и, в случае необходимости, восстанавливать исходное состояние.

- Show state(Показать состояние) – показывает текущее состояние концентратора, может принимать два значения normal(концентратор работает) и freezed(концентратор был остановлен из-за ошибки).
- Reset(Перезагрузить) – если концентратор находится в состоянии останова, то эта команда вернет его в рабочее состояние.

В меню устройств работающих на сетевом уровне (маршрутизаторы и компьютеры) в дополнение к основным имеются ещj шесть пунктов:

- Set TCP/IP Properties(Установка свойств TCP/IP) – вызывает диалог позволяющий изменить

свойства TCP/IP;

•Send Ping...(Послать эхо-запрос) – позволяет послать эхо-запрос адресату;

•ARP – подменю позволяет работать с таблицей протокола ARP на выбранном устройстве;

•Counters – подменю содержит два пункта:

•Show Packet Counters(Показать счетчики пакетов) – показывает счетчики для пакетов протоколов ARP, IP, UDP, TCP

•Reset Packet Counters(Сбросить счетчики пакетов) – устанавливает все счетчики на выбранном устройстве в ноль;

•Console – вызывает командную строку, позволяющую настраивать таблицы маршрутизации, ARP таблицы и др.;

•Print route table(Показать таблицу маршрутизации) – выводит на вкладку "консоль" таблицу маршрутизации выбранного сетевого устройства.

Пункт контекстного меню Send Ping...(Послать эхо-запрос) – вызывает диалог, в котором можно настроить параметры эхо-запроса. Во время передвижения пакетов по сети во вкладке "консоль" должны появится сообщения, аналогичные приведенным ниже:

```
PC1 Echo Request Packet Network Created Echo

PC1 Echo Request Packet Network Created Echo

          Request packet to 10.0.0.2

...

PC1 Echo Reply Packet Network Echo reply packet

          received from 10.0.0.2
```

Меню ARP позволяет управлять таблицей протокола ARP на выбранном устройстве содержит три подпункта:

•Add static entry to ARP table – вызывает два диалоговых окна: в первом вводится MAC адрес, а во втором IP адрес, после чего в ARP таблицу заноситься статическая запись о связи IP и MAC адресов;

•Remove entry from ARP table – вызывает диалоговое окно, позволяющее ввести IP адрес, для которого будет удалена запись из ARP таблицы;

•Print ARP table – выводит на вкладку "консоль" ARP таблицу выбранного сетевого устройства.

В контекстное меню компьютеров, т.е. устройств поддерживающих уровень приложений добавляется еще один пункт: подменю Applications(Приложения), которое позволяет работать с протоколами: Echo(UDP,TCP), SNMP и TELNET.

(начало)

5.4. Командная строка

Для запуска командной строки из контекстного меню выберем "Console", появится окно консоли (рис. 5.4).

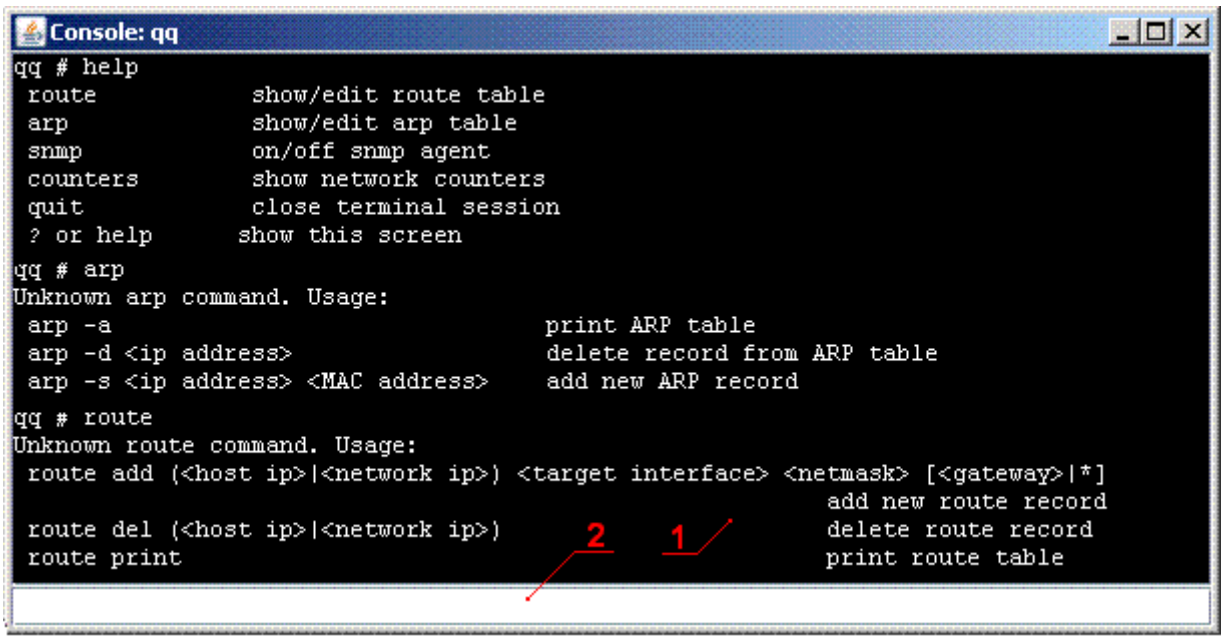

Рис. 5.4. Консоль.

Окно разделено на 2 части:

- область для сохранения результата выполнения команд (1);
- командная строка, в которой можно вводить команды на выполнение (2).

В консоли могут использоваться следующие специальные клавиши:

- <Enter> – выполнить введенную команду;
- стрелки вверх/вниз – просмотр истории команд;
- <ESC> – очистить командную строку;
- Ctrl+D – закрыть консоль.

В командной строке с помощью команды route можно выполнить настройку статической таблицы маршрутизации. Для этого предназначена команда route.

Описание синтаксиса команды route:

- route add (<ip адрес устройства>|<ip адрес сети>) <интерфейс> <маска подсети> [<шлюз>| *] – добавить новый маршрут для сети или устройства;
- route del (<ip адрес устройства>|<ip адрес сети>) – удалить существующий маршрут для указанного IP адреса устройства или сети;
- route print – просмотреть список существующих маршрутов.

С помощью команды arp можно выполнить настройку таблицы ARP:

- arp -a просмотреть ARP таблицу;
- arp -d <ip address> удалить из ARP таблицы запись об IP адресе;
- arp -s <ip address> <MAC address> добавить ARP запись связывающую IP и MAC адреса.

С помощью команды snmp можно управлять snmp агентом:

- snmp (on|<port number>) [community name] включить SNMP агента. Если порт не указан (значение on), то по умолчанию выбирается порт 161. Если не указано имя группы доступа, то берется значение по умолчанию public;
- snmp off выключить SNMP агента.

При вводе команд route, arp и snmp без параметров будет выведена краткая информация по их использованию.

(начало)

5.5. Работа с протоколами уровня приложений

В имитаторе javaNetSim имеется возможность работы со следующими протоколами уровня приложений стека протоколов TCP/IP:

- Echo(UDP и TCP реализации),
- SNMP и TELNET.

(начало)

5.5.1 Работа с протоколом Echo

Имитатор javaNetSim позволяет использовать протоколы UDP или TCP в качестве транспортных протоколов для протокола Echo. Для установки echo-сервера в режим прослушивания порта в контекстном меню надо выбрать пункт:

- "Applications" -> "Start udp echo server to listen" - для Echo-UDP
- "Applications" -> "Start tcp echo server to listen" - для Echo-TCP.

После этого в появившемся диалоговом окне следует вести номер порта, на котором выбранное приложение будет ожидать сообщения. После этого с любого другого узла можно отсылать сообщения на тот узел, на котором запущен echo-сервер и получать ответы.

Для того, чтобы послать эхо-запрос, необходимо в контекстном меню выбрать

- "Applications" -> "Send data via udp echo client" - для Echo-UDP
- "Applications" -> "Send data via tcp echo client" - для Echo-TCP

и ввести четыре параметра:

- IP-адрес компьютера, на котором запущен echo-сервер;
- номер порта на котором echo-сервер ожидает сообщения;
- сообщение – любой текст;
- количество посылаемых сообщений, т.е. количество копий сообщения отправляемых echo-серверу.

Протокол Echo обладает простой структурой, поэтому при помощи telnet- клиента можно подключиться к Echo-TCP-серверу. В таком режиме нажатие любой клавиши на клавиатуре будет сопровождаться выводом ее на экран терминала.

(начало)

5.5.2. Работа с протоколом SNMP

В имитаторе javaNetSim предусмотрено несколько функций для работы с протоколом SNMP:

- запуск SNMP агента на объекте управления;
- остановка SNMP агента на объекте управления;
- посылка SNMP запросов агенту.

Для запуска SNMP агента необходимо выбрать пункт контекстного меню "Application" -> "Start SNMP Agent" и задать два параметра:

- порт, на котором SNMP агент будет ожидать пакеты;
- имя группы доступа для SNMP агента.

Для остановки SNMP агента необходимо выбрать пункт контекстного меню "Application" -> "Stop SNMP Agent".

Для того, чтобы послать запрос SNMP агенту необходимо выбрать пункт контекстного меню "Application" -> "Send SNMP message" и заполнить поля диалога, приведенные на рис. 5.5.

- IP Address – IP адрес компьютера на котором установлен SNMP агент.
- Destination Port – порт на котором SNMP агент ожидает пакеты.
- SNMP message – SNMP запрос, может принимать значения: get, getnext, set.
- Variables – SNMP переменные описываемые деревом MIB.
- Community name – имя группы доступа, которое должно совпадать с именем группы доступа установленным при создании агента.

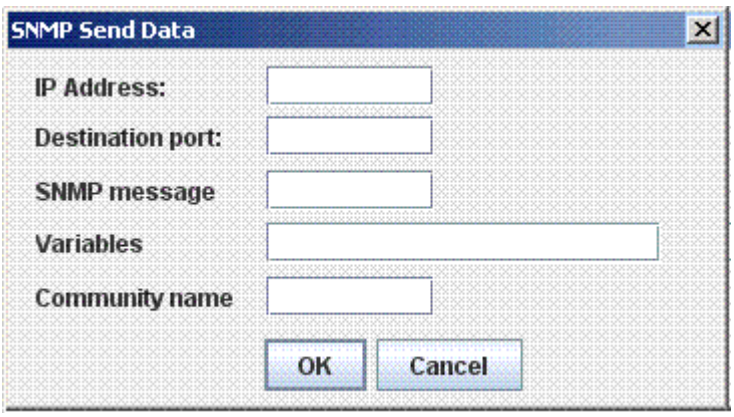

Рис. 5.5. Создание SNMP запроса.

Поле Variables имеет специальный формат, различный для запросов get(getnext) и set. Если SNMP запрос является get или getnext запросом, то строка переменных должна выглядеть следующим образом:

```
<переменная> [;<переменная>]

Например: ip.address_eth0;device.hostname.
```

А если SNMP запрос является set запросом, то в строке переменных к каждой переменной добавляется значение:

```
<переменная>="<значение>" [;<переменная>="<значение>"]

Например: ip.address_eth0="192.168.10.3"
```

Результаты запроса будут выведены на вкладку "консоль". Например:

```
PC2 SNMP Protocol Data Application Received getResponse:

'IP.Address_Eth0=172.168.0.2' , 'Device.Hostname=PC1'
```

Список SNMP переменных, поддерживаемых имитатором javaNet- Sim, которые имеют режим доступа "только для чтения"приведен ниже.

- Counter.InputIP – количество пришедших IP пакетов;
- Counter.OutputIP – количество отправленных IP пакетов;
- Counter.ARP – количество обработанных ARP пакетов;
- Counter.InputTCP – количество пришедших TCP пакетов;
- Counter.OutputTCP – количество отправленных TCP пакетов;
- Counter.ReceiveDuplicatedTCP – количество дублирующихся пакетов TCP полученных устройством;
- Counter.SendDuplicatedTCP – количество дублирующихся пакетов TCP отправленных устройством;
- Counter.SendAckTCP – количество посланных ACK пакетов;
- Counter.InputUDP – количество пришедших UDP пакетов;
- Counter.OutputUDP – количество отправленных UDP пакетов;
- Device.AllInterfaces – список всех возможных интерфейсов устройства;
- Device.AvailableInterfaces – список всех доступных интерфейсов устройства;
- Device.Hostname – имя устройства;
- Device.MACaddress_Eth0 – MAC адрес устройства на интерфейсе Ethernet0;
- IP.AllInterfaces – список всех возможных интерфейсов устройства работающих по протоколу IP;
- IP.ARPTable – ARP таблица для устройства;
- SNMP.revision – версия модификации SNMP;
- SNMP.version – версия SNMP.

Некоторые SNMP переменные имеют режим доступа "чтение и запись".

- IP.DefaultGateway – шлюз по умолчанию;
- IP.Address_Eth0 – IP адрес интерфейса Ethernet0;
- IP.SubnetMask_Eth0 – маска интерфейса Ethernet0;
- SNMP.CommunityName – имя группы доступа для SNMP агента.

Режим доступа определяет действия, которые можно производить с переменной. Если переменная имеет

режим доступа только чтение, то попытка записать новое значение завершиться с ошибкой.

(начало)

5.5.3. Работа с протоколом TELNET

В имитаторе javaNetSim предусмотрены следующие функции для работы с протоколом TELNET:

- запуск TELNET сервера на управляемом компьютере;
- остановка TELNET сервера;
- запуск TELNET клиента.

Для запуска TELNET сервера необходимо выбрать пункт контекстного меню "Application" -> "Start telnet server to listen" и задать два параметра:

- порт, на котором TELNET-сервер будет ожидать пакеты;
- пароль для доступа к TELNET-серверу.

Для остановки TELNET сервера необходимо выбрать пункт контекстного меню "Application" -> "Stop telnet server".

Для соединения с TELNET сервером необходимо выбрать пункт контекстного меню "Application" -> "Telnet client" и задать два параметра:

- IP адрес TELNET-сервера;
- порт, на котором TELNET-сервер ожидает пакеты.

После этого откроется окно терминала и если соединение прошло успешно появится приглашение ввести имя пользователя: login. После введения имени появится приглашение ввести пароль: password. После введения пароля, имя пользователя и пароль проверяются и, если они корректны, будет выведено приглашение в виде:

```
<имя компьютера> #
```

В javaNetSim для TELNET-сервера используется имя пользователя root и пароль, установленный при создании TELNET-сервера. В сеансе telnet доступны следующие команды:

- route – просмотр и редактирование сетевых маршрутов;
- arp – просмотр и редактирование ARP таблиц;
- snmp – запуск о остановка SNMP агента;
- counters – просмотр доступных сетевых счетчиков;
- passwd – изменение пароля на доступ к TELNET серверу;
- quit – закрыть TELNET сеанс;
- ? или help – посмотреть список доступных команд.

После завершения работы необходимо закрыть сеанс telnet. Закрытие сеанса telnet можно произвести тремя способами:

- набрать команду quit.
- нажать комбинацию клавиш Ctrl+D.
- просто закрыть окно терминала.

Несмотря на то, что протокол TELNET в javaNetSim реализован на очень простом уровне, это не мешает ему выполнять свои функции. В качестве примера можно привести подключения telnet-клиента к Echo-TCP-серверу.