Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Сети и телекоммуникации

Отчет по лабораторной работе
по сетевым технологиям

Работу
выполнила:
Михалёва М.В.
Группа: 43501/1
Преподаватель:
Алексюк А.О

Санкт-Петербург
2018

Содержание

1. Цель работы

Ознакомиться с принципами программирования собственных протоколов, созданных на основе TCP и UDP.

2. Краткое описание выполненных базовых работ по TCP и UDP

В ходе выполнения лабораторных работ были написаны простейшие клиент-серверные приложения на базе протоколов TCP и UDP.

2.1. TCP

В приложениях TCP создается сокет, ставиться на прослушивание и при подключении клиента создается отдельный сокет, по которому клиент общается с сервером.

Для инициализации, запуска и завершения TCP-сервера необходимо выполнить следующие системные вызовы:

- socket() - создание сокета
- bind() - привязка созданного сокета к заданным IP-адресам и портам
- listen() – перевод сокета в состояние прослушивания
- accept() - прием поступающих запросов на подключение и возврат сокета для нового соединения
- recv() - чтение данных от клиента из сокета, полученного на предыдущем шаге
- send() - отправка данных клиенту с помощью того же сокета
- shutdown() - разрыв соединения с клиентом
- close() - закрытие клиентского и слушающего сокетов

TCP-клиенты выполняют следующую последовательность действий для открытия соединения, отправки и получения данных, и завершения:

- socket() - создание сокета
- connect() - установка соединения для сокета, который будет связан с серверным сокетом, порожденным вызовом accept()
- send() - отправка данных серверу
- recv() - прием данных от сервера
- shutdown() - разрыв соединения с сервером
- close() - закрытие сокета

Так же был реализован сервер, поддерживающий работу с несколькими клиентами.

Для этого при подключении клиента создается поток, который в котором создается сокет для общения с клиентом.

2.2. UDP

В приложениях UDP сервер принимает сообщение от клиента и отправляет сообщение об успешной доставке. UDP протокол не подразумевает логических соединений, поэтому не создается слушающего сокета.

Реализация UDP-сервера имеет следующий вид:

- socket() - создание сокета
- bind() - привязка созданного сокета к заданным IP-адресам и портам
- recvfrom() - получение данных от клиента, параметры которого заполняются функцией
- sendto() - отправка данных с указанием параметров клиента, полученных на предыдущем шаге
- close() - закрытие сокета

UDP-клиент для обмена данными с UDP-сервером использует следующие функции:

- socket() - создание сокета
- recvfrom() - получение данных от сервера, параметры которого заполняются функцией
- sendto() - отправка данных с указанием параметров сервера, полученных на предыдущем шаге
- close() - закрывает сокет

3. Индивидуальное задание

Задание: разработать приложение-сервер «Удаленный калькулятор», позволяющее по запросу выполнять математические операции, и удаленный клиент для сервера.

Основные возможности. Серверное приложение должно реализовывать следующие функции:

1. Прослушивание определенного порта
2. Обработка запросов на подключение по этому порту от клиентов
3. Поддержка одновременной работы нескольких клиентов через механизм нитей
4. Приём «быстрых» операций с аргументами от клиента. Должны поддерживаться следующие операции: сложение, вычитание, умножение, деление
5. Вычисление «долгих» математических операций (факториал, квадратный корень) с последующей отложенной посылкой результата клиенту (отдельная операция, инициируемая сервером)
6. Обработка запроса на отключение клиента
7. Принудительное отключение клиента

Клиентское приложение должно реализовывать следующие функции:

1. Установление соединения с сервером
2. Посылка операции с аргументами на вычисление
3. Получение результата вычислений «быстрых» операций
4. Получения результата вычислений «долгих» операций
5. Разрыв соединения
6. Обработка ситуации отключения клиента сервером

Настройки приложений. Разработанное клиентское приложение должно предоставлять пользователю настройку IP-адреса или доменного имени удалённого калькулятора и номера порта, используемого сервером.

Разработанное серверное приложение должно предоставлять пользователю настройку времени выполнения «долгих» операций.

Методика тестирования. Для тестирования приложений запускается сервер «Удаленного калькулятора» и несколько клиентов. В процессе тестирования проверяются основные возможности калькулятора по мгновенному и отложенному выполнению удалённых операций.

4. Разработанный прикладной протокол

Возможные команды клиента:

1. "Быстрые"операции
   (a) Сложение (+)
   (b) Вычитание (-)
   (c) Умножение (*)
   (d) Деление (/)

   Результат таких операций приходит мгновенно. Выполняются для двух переменных:
   First operand: <op1>
   Second operand: <op2>
2. "Долгие"операции
   (a) Факториал (!)
   (b) Квадратный корень (sqrt)

   Результат приходит за случайное время (от 1 до 10 секунд). Выполняются для одной переменной: Operand: <op>
3. Задание id клиента при запуске: ./client <id>
4. Отключение клиента: quit

Возможные команды клиента:

1. Получения списка id подключенных клиентов: ls

2. Отключение клиента по id: disconnect <id>

Возможные ответы сервера:

1. Результат "быстрой"операции: <op1> <cmd> <op2> = <res>
2. Результат "долгой"операции: Long operation result: (processing time = <1:10s>) <op> <cmd> = <res>
3. Сообещние об ожидании результата "долгой"операции и возможности продолжить расчеты: The result will be recieved after few seconds. You can use other commands.
4. Вывод сообщения о принудительном отключении: Sorry! You were disconnected. Bye.

5. Описание архитектуры и особенности реализации TCP

Протокол TCP выполняет функции транспортного уровня (transport layer) и обеспечивает надежную службу пересылки данных для приложений. В TCP/IP встроен специальный механизм, гарантирующий пересылку данных без ошибок и пропусков и в той последовательности, в которой они были отправлены.

Приложения, например пересылки файлов, передают данные в TCP, который добавляет к ним заголовок и формирует элемент, называемый сегментом (segment).

TCP отсылает сегменты в IP, в котором производится маршрутизация данных в заданное место. На другой стороне соединения TCP предполагает получение тех же сегментов данных от IP, определяет приложение, которому направлены эти данные, и передает их приложению в том порядке, в котором они были отправлены.

6. Описание архитектуры и особенности реализации UDP

Приложение может послать другому приложению независимое сообщение с помощью протокола UDP, который добавляем к сообщению заголовок и формирует элемент, называемый датаграммой UDP или сообщением UDP.

UDP передает исходящие сообщения в IP и предполагает на другой стороне получение входящих сообщений от IP. Далее UDP определяет приложение, которому направлены данные.

UDP реализует коммуникационную службу без создания соединения, которая часто используется для просмотра содержимого простых баз данных.

7. Тестирование приложения на основе TCP

При тестировании запускался сервер и несколько клиентов.
При подключении клиента у нас есть возможность задать порт и/или IP.

Листинг 1: Подключение клиента

```
1 maria_rheon@Rheon:/Documents/Calculator_tcp/mary.mikhaleva/client$ ./client
2 Use "./client <host_ip> <host_port>"
```

Без явного указания IP = 127.0.0.1, порт = 7000.
При запуске клиентов укажем им id.

Листинг 2: id клиентов

```
maria_rheon@Rheon:~/Documents/Calculator_tcp/mary_mikhaleva/mary.mikhaleva/tcp/
    ↪ client$ ./client me
Connected successfully client:
Enter command (+,-,*,/,!,sqrt):
--------
maria_rheon@Rheon:~/Documents/Calculator_tcp/mary_mikhaleva/mary.mikhaleva/tcp/
    ↪ client$ ./client another
Connected successfully client:
Enter command (+,-,*,/,!,sqrt):
```

Проверим список подключений на сервере.

Листинг 3: Список клиентов

```
maria_rheon@Rheon:~/Documents/Calculator_tcp/mary_mikhaleva/mary.mikhaleva/tcp/
    ↪ server$ ./server
Enter command: ls
me
another
```

Теперь попробуем найти сумму, факториал и разность. При этом последнее начнем выполнять сразу после запроса результата факториала. В конце отключимся от сервера.

Листинг 4: Тестирование операций

```
maria_rheon@Rheon:~/Documents/Calculator_tcp/mary_mikhaleva/mary.mikhaleva/tcp/
    ↪ client$ ./client me
Connected successfully client:
Enter command (+,-,*,/,!,sqrt): +
First operand: 3
Second operand: 4
3 + 4 = 7
Enter command (+,-,*,/,!,sqrt): !
Operand: 4
Command was sent to server
The result will be received after few seconds
You can use other commands while waiting
Enter command (+,-,*,/,!,sqrt): -
First operand: 3
Second operand: 6
3 - 6 = -3
Enter command (+,-,*,/,!,sqrt):
Long operation result (processing time = 8s): 4! = 24
Enter command (+,-,*,/,!,sqrt): quit
Bye!
```

Попробуем в качестве команды и операндов вводить некорректные данные.

Листинг 5: Тестирование ошибочных данных

```
maria_rheon@Rheon:~/Documents/Calculator_tcp/mary_mikhaleva/mary.mikhaleva/tcp/
    ↪ client$ ./client me
Connected successfully client:
Enter command (+,-,*,/,!,sqrt): +
First operand: 3
Second operand: 4
3 + 4 = 7
Enter command (+,-,*,/,!,sqrt): !
```

```
Operand: 4
Command was sent to server
The result will be received after few seconds
You can use other commands while waiting
Enter command (+,-,*,/,!,sqrt): -
First operand: 3
Second operand: 6
3 - 6 = -3
Enter command (+,-,*,/,!,sqrt):
Long operation result (processing time = 8s): 4! = 24
Enter command (+,-,*,/,!,sqrt): quit
Bye!
```

Отключим одного из подключенных клиентов.

Листинг 6: Тестирование ошибочных данных

```
maria_rheon@Rheon:~/Documents/Calculator_tcp/mary_mikhaleva/mary.mikhaleva/tcp/
    ↪ server$ ./server
Enter command: ls
me
Enter command: disconnect me

------
Enter command (+,-,*,/,!,sqrt):
Sorry! You were disconnected. Bye.
```

8. Тестирование приложения на основе UDP

Проведем тестирование аналогично предыдущему. Отличие в том, что при подключении каждому клиенту задается свой порт. При любом выполнении операции порт текущего клиента выводится в сервере. Для отключения клиента нужно ввести его порт в терминале. Сообщение об отключении появится при попытке выполнить операцию. Приведем пример выполнения с двумя клиентами, в одном из которых будем вводить ошибочные команды.

Листинг 7: Тестирование калькулятора на основе UDP

```
maria_rheon@Rheon:~/Documents/Calculator_udp/withClose/CloseMessage/udp$ ./
    ↪ server
Endpoint created
Address and port assigned to socket
41649 // 1+2
41649 // 7!
55392 // 3+5
41649 //0+0
55392 //43*4
55392 //disconnect client 2

----client 1
maria_rheon@Rheon:~/Documents/Calculator_udp/withClose/CloseMessage/udp$ ./
    ↪ client
Input command (+, -, *, /, !, sqrt, quit): +
Operand 1: 1
Operand 2: 2
1 + 2 = 3
Input command (+, -, *, /, !, sqrt, quit): !
Operand: 7
Input command (+, -, *, /, !, sqrt, quit):
```

```
7! = 5040
You may enter another operation: +
Operand 1: 0
Operand 2: 0
0 + 0 = 0
Input command (+, -, *, /, !, sqrt, quit): quit
Bye
----client 2
maria_rheon@Rheon:~/Documents/Calculator_udp/withClose/CloseMessage/udp$ ./
    ↪ client
Input command (+, -, *, /, !, sqrt, quit): fdfg
Unknown command
Please, use one of the following: +, -, *, /, !, sqrt, quit
Input command (+, -, *, /, !, sqrt, quit): +
Operand 1: rt
You typed not a number; please, try to type the number
Operand 1: 3
Operand 2: 5
3 + 5 = 8
Input command (+, -, *, /, !, sqrt, quit): *
Operand 1: 43
Operand 2: o
You typed not a number; please, try to type the number
Operand 2: 4
43 * 4 = 172
Input command (+, -, *, /, !, sqrt, quit): /
Operand 1: 2
Operand 2: 4
You were disconnected) Bye
```

9. Листинги программ

Листинги можно найти в репозитории на GitHub:
https://github.com/mariarheon/NetworksLab2018/tree/ind_task/individual_task

10. Дополнительное задание

10.1. Подключение к HTTP-серверу и запрос веб-страницы

Запускаем программу telnet. Указываем ей, что хотим подключиться к хосту с доменным именем ng.ru и использовать для подключения 80 TCP-порт.

```
maria_rheon@Rheon:~$ telnet ng.ru 80
```

telnet определяет IP-адрес хоста с таким доменным именем по протоколу DNS. Оказывается, что хост с именем ng.ru имеет IP-адрес 173.194.122.130. Далее telnet пробует инициировать TCP-подключение к машине с этим IP-адресом (connect()).

```
Trying 188.40.89.58...
```

После того, как подключение успешно выполнилось, telnet сообщает нам об этом.

```
Connected to ng.ru.
Escape character is '^]'.
```

Вводим сообщение, которое необходимо передать серверу ng.ru на 80 порт (хост ожидает, что мы будем поверх TCP-протокола использовать HTTP-протокол (сервера, работающие на таком протоколе, обычно ожидают подключения клиентов на 80 TCP-порту)).
В соответствии с HTTP, для получения веб-страницы с url = domain.name/some/url, необходимо в качестве первой строки передать следующее:
GET / HTTP/1.1
где GET - име метода, который трактуется HTTP-сервером так же, как если бы мы ввели url в обычном броузере и нажали бы <Enter> В конце каждой строки передаются 2 символа: ’̊’, ’ n’, что означает конец строки.

```
GET / HTTP/1.0
HOST: ng.ru
```

Сервер обрабатывает запрос и отвечает нам. При этом telnet отображает его ответ.
В соответствии с протоколом HTTP, сервер в первой строке ответа должен передать следующую информацию: <Версия протокола HTTP> <HTTP-статус> <Текстовое пояснение HTTP-статуса>
В данном случае мы видим следующую надпись:
HTTP/1.1 301 Moved Permanently которая означает, что сервер нам ответил, используя версию 1.1 протокола HTTP. Статус ответа = 301, что означает, что ресурс был перемещен на постоянной основе в новое месторасположение.

```
HTTP/1.1 301 Moved Permanently
Server: nginx
Date: Thu, 20 Dec 2018 20:02:35 GMT
Content-Type: text/html
Content-Length: 178
Connection: close
Location: http://www.ng.ru/

<html>
<head><title>301 Moved Permanently</title></head>
<body bgcolor="white">
<center><h1>301 Moved Permanently</h1></center>
<hr><center>nginx</center>
</body>
</html>
Connection closed by foreign host.
```

Повторим эксперимент для сайта кафедры КСПТ. Заметим, что HTTP-статус = 200, что соответствует корректной обработке запроса.

Листинг 8: Корректный запрос веб-страницы

```
=======
maria_rheon@Rheon:~$ telnet kspt.icc.spbstu.ru 80
Trying 195.209.231.146...
Connected to kspt.icc.spbstu.ru.
Escape character is '^]'.
GET / HTTP/1.0
Host: kspt.icc.spbstu.ru

HTTP/1.1 200 OK
Date: Thu, 20 Dec 2018 19:58:43 GMT
Server: Apache/2.2.22 (Mandriva Linux/PREFORK-0.1mdv2010.2)
Vary: Cookie
Content-Length: 8920
```

```
Content-Type: text/html; charset=utf-8
Connection: close


<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title КСПТ>: Официальный сайт кафедры КСПТ</title>

    <link rel="stylesheet" type="text/css" href="/media/css/new/main.css"/>
    <link rel="icon" type="image/png" href="/media/image/icon.png"/>
  </head>
  <body>
    <div id="headline" class="header">
      <div id="headline-right"></div>
      <div id="headline-left">
        <div id="logo">
          <a href="/"><img src="/media/image/aivt_logo.gif" alt="Кафедра АиВТ
    ↪ СПбГПУ"/></a>
        </div>
      </div>
    </div>
    <div id="menu" class="nav">
      <ul>

  <li class="active">
      <a href="/"Главная></a>

  </li>
  <li>
      <a href="/info/"Кафедра></a>

  </li>
.........


  <h1Кафедра> компьютерных систем и программных технологий</h1>
<img src="/media/files/photo/pictures/9.jpg" width="400"  class="photo float-
    ↪ left" alt="Фото" />

<pКафедра> компьютерных систем и программных технологий (<a href="http://aivt.ftk.
    ↪ spbstu.ru/news/2009/12/08/%D0%BD%D0%BE%D0%B2%D0%BE%D0%B5-%D0%BD%D0%B0%D0%
    ↪ B7%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BA%D0%B0%D1%84%D0%B5%D0%B4%D1%80%D1%8
    ↪ B/"до> 2009 года</a> — кафедра автоматики и вычислительной техники АиВТ) ведет
    ↪ свою <a href="/info/history/"историю></a> с 1933 г. и в настоящее время
    ↪ входит в состав <a href="http://icc.spbstu.ru" target="blank"Института>
    ↪ компьютерных наук и технологий</a> ИКНТ(, до 2015 г – Институт
    ↪ информационных технологий и управления ИИТУ, ранее – Факультет технической
    ↪ кибернетики ФТК) СанктПетербургского– политехнического университета Петра
    ↪ Великого (<a href="http://spbstu.ru" target="blank"СПбПУ></a>).</p>
<pКафедра> готовит бакалавров и магистров по направлению <a href="/education/degrees/
    ↪ "«Информатика> и вычислительная техника»</a>. </p>
<pВ> <a href="/info/staff/"составе> кафедры</a> 3 профессора, доктора наук и более 25
    ↪  доцентов, кандидатов наук.</p>
<pКафедра> имеет 15 <a href="/education/labs/"учебных> лабораторий</a>.</p>
<p><a href="/research/science/"Научные> направления</a> связаны с проектированием
    ↪ программного обеспечения, разработкой компьютерных систем и систем управления.</p>
<pЗаведующий> кафедрой – кандидат технических наук, доцент <a href="/info/staff/
    ↪ itsykson/"ВМ>.. Ицыксон</a>.</p>
<pНаучный> руководитель кафедры — заслуженный работник высшей школы РФ, доктор
    ↪ технических наук, профессор <a href="/info/staff/melehin/"ВФ>.. Мелехин</a
```

```
    ↪ >.</p>
58   <div class="article-meta">
59    ......
60            <p class="copyright vcard">&copy; 1998/2009 <a href="http://ftk.spbstu.ru/
     ↪ structure/centers/tcc/" class="url fn org">ТКЦ> ФТК</a></p>
61
62        </div>
63        <div class="clear"></div>
64     </div>
65
66     <script src="http://www.google-analytics.com/ga.js" type="text/javascript"
     ↪ ></script>
67     <script type="text/javascript">
68        try {
69          var pageTracker = _gat._getTracker("UA-10614285-1");
70        } catch (err) {}
71     </script>
72
73   </body>
74 </html>
75
76 Connection closed by foreign host.
```

10.2. Подключение к FTP-серверу, запрос списка файлов в директории и получение файла

Для анализа работы по протоколу FTP воспользуемся ftp-сервером mirror.yandex.ru. Подключаемся к этому серверу (FTP использует порт 21).

```
1 maria_rheon@Rheon:~$ telnet mirror.yandex.ru 21
2 Trying 213.180.204.183...
3 Connected to mirror.yandex.ru.
4 Escape character is '^]'.
```

В ответ приходит следующее:

```
1 220-Welcome to Yandex Mirror FTP service. Your served by: node01e.mirror.yandex.
    ↪ net
```

220 - это код ответа от сервера, далее пояснительное сообщение. Код 220 значит, что всё хорошо.

Авторизуемся как анонимный пользователь:

```
1 220
2 user anonymous
3 331 Please specify the password.
4 pass mikhspbstu@mail.ru
5 230 Login successful.
```

Здесь в качестве логина указываем "anonymous a в качестве пароля "mikhspbstu@mail.ru". Сервер нас пускает.

Выполняем переход в пассивный режим, чтобы просмотреть содержимое текущего каталога ("/"):

```
1 pasv
2 227 Entering Passive Mode (213,180,204,183,224,84)
```

Здесь очевидно, что 6 чисел означают IP-адрес и порт, куда мы должны параллельно подключиться (используя протокол TCP), чтобы получить ответ от команды, требующей

пассивного режима. Первые 4 числа означают IP-адрес хоста, который ожидает нас принять. Т.о. IP-адрес хоста = 213.180.204.183. Последние 2 числа при совмещении дают порт, на котором нас ожидают: (224«8) + 84 = 57428.

Далее, не выходя из текущей сессии, создаём новую сессию, указав тот IP-адрес и порт:

```
maria_rheon@Rheon:~$ telnet 213.180.204.183 57428
Trying 213.180.204.183...
Connected to 213.180.204.183.
Escape character is '^]'.
```

В первой сессии, указываем, что хотим получить содержимое текущего каталога ("/"), и оно будет отправлено клиенту второй сессии:

```
pasv
227 Entering Passive Mode (213,180,204,183,224,84)
list
150 Here comes the directory listing.
226 Directory send OK.
```

После выполнения этой команды в первой сессии, во второй имеем информацию о лежащих внутри каталога ("/") файлах и подкаталогов:

```
drwxr-xr-x   21 ftp      ftp          4096 Dec 20 04:46 altlinux
drwxr-sr-x   21 ftp      ftp          4096 Dec 06 04:15 altlinux-beta
drwxr-xr-x    9 ftp      ftp          4096 Dec 20 15:14 altlinux-nightly
drwxr-xr-x    5 ftp      ftp          4096 Dec 12 19:46 altlinux-starterkits
drwxr-xr-x   17 ftp      ftp          4096 Dec 20 18:34 archlinux
drwxr-xr-x    7 ftp      ftp          4096 Oct 12  2015 archlinux-arm
drwxr-xr-x    8 ftp      ftp          4096 Dec 20 19:32 archlinux32
dr-xr-xr-x    7 ftp      ftp          4096 Dec 06 03:12 archserver
drwxr-xr-x    4 ftp      ftp          4096 Dec 12 15:46 astra
drwxr-xr-x   10 ftp      ftp          4096 Dec 20 17:56 calculate
drwxrwxr-x   48 ftp      ftp          4096 Dec 20 18:09 centos
drwxr-xr-x    9 ftp      ftp          4096 Dec 20 18:52 debian
drwxrwsr-x    5 ftp      ftp          4096 Nov 20 15:22 debian-backports
drwxr-xr-x    5 ftp      ftp          4096 Dec 20 15:36 debian-cd
drwxr-sr-x    8 ftp      ftp          4096 Dec 20 02:20 debian-multimedia
drwxr-xr-x   19 ftp      ftp          4096 Dec 20 17:43 debian-ports
drwxr-xr-x    7 ftp      ftp          4096 Dec 20 18:05 debian-security
drwxrwsr-x    7 ftp      ftp          4096 Dec 19 05:32 epel
drwxr-xr-x    9 ftp      ftp          4096 Dec 20 17:54 fedora
drwxrwxr-x    5 ftp      ftp          4096 Dec 20 17:50 fedora-secondary
drwxrwxr-x    7 ftp      ftp          4096 Dec 20 17:16 freebsd
drwxr-xr-x    6 ftp      ftp          4096 Dec 20 19:32 gentoo-distfiles
drwxr-xr-x  171 ftp      ftp          4096 Dec 20 20:02 gentoo-portage
drwxr-xr-x   10 ftp      ftp         12288 Dec 05 21:02 knoppix
drwxr-xr-x    3 ftp      ftp          4096 Dec 20 15:06 libreoffice
drwxr-xr-x    5 ftp      ftp          4096 Dec 18 13:34 linuxmint
drwxr-xr-x    5 ftp      ftp          4096 Dec 20 11:16 linuxmint-packages
drwxr-xr-x    6 ftp      ftp          4096 Dec 20 17:26 macports
drwxr-xr-x    6 ftp      ftp          4096 Dec 20 20:10 mageia
drwxr-xr-x   53 ftp      ftp          4096 Sep 21 15:11 mirrors
drwxr-xr-x    6 ftp      ftp          4096 Mar 11  2014 mopslinux
drwxr-xr-x   58 ftp      ftp          4096 Dec 20 15:23 openbsd
drwxr-xr-x    5 ftp      ftp          4096 Dec 20 19:06 openmandriva
drwxr-xr-x    8 ftp      ftp          4096 Dec 20 20:22 opensuse
drwxr-xr-x    9 ftp      ftp          4096 Dec 20 19:05 pub
drwxr-xr-x   36 ftp      ftp          4096 Dec 18 15:33 puias
drwxr-xr-x   15 ftp      ftp          4096 Dec 10 21:00 puppyrus
drwxrwxrwx   14 ftp      ftp          4096 Dec 20 18:42 rosa
drwxr-xr-x    6 ftp      ftp          4096 Dec 11 21:00 runtu
```

```
drwxrwxr-x    9 ftp      ftp          4096 Dec 20 17:35 sabayon
drwxr-xr-x   27 ftp      ftp          4096 Dec 20 17:34 scientificlinux
drwxr-xr-x    6 ftp      ftp          4096 Apr 02  2014 simplelinux
drwxr-xr-x   49 ftp      ftp          4096 Dec 20 17:35 slackware
drwxr-xr-x   15 ftp      ftp          4096 Dec 15 21:39 slackwarearm
drwxr-xr-x    7 ftp      ftp          4096 Dec 20 18:54 ubuntu
drwxr-xr-x   28 ftp      ftp          4096 Dec 20 20:16 ubuntu-cdimage
drwxr-xr-x    6 ftp      ftp          4096 Dec 20 15:50 ubuntu-ports
drwxr-xr-x   13 ftp      ftp          4096 Dec 20 19:16 ubuntu-releases
Connection closed by foreign host.
```

В первой сессии перейдем в нужную нам директорию и выберем на загрузку README.txt

```
cwd /slackware/slackware64
250 Directory successfully changed.
retr README.TXT
150 Opening BINARY mode data connection for README.TXT (8564 bytes).
226 Transfer complete.
```

Во второй получим загруженный нами файл:

```
Welcome to Slackware 14.2!  (64-bit x86_64 edition)

Slackware 14.2 is a complete distribution of the Linux operating system.

Here are some versions of major components of Slackware 14.2:

- Linux kernel            4.4.14
- C compiler              gcc-5.3.0
- Binutils                2.26
- GNU C Library           glibc-2.23
- X Window System         X11R7.7
- KDE                     4.14.21 (KDE 4.14.3 with kdelibs-4.14.21)
- Xfce                    4.12.1

For installation instructions, see the file 'Slackware-HOWTO'.

For important hints about this release, see the file 'CHANGES_AND_HINTS.TXT'.

These are some of the important files and directories found on the Slackware
FTP site:

  ftp://ftp.slackware.com/pub/slackware/slackware64-14.2/

Thanks to cwo.com for continuing to help us with hosting for our web site,
and to the OSU Open Source Lab for hosting our archives at ftp.slackware.com
(aka ftp.osuosl.org).

If you're reading this on a CD-ROM, these directories will probably be
split across several discs.

........

Patrick Volkerding
volkerdi@slackware.com

Connection closed by foreign host.
```

10.3. Подключение к SMTP-серверу и отправка письма

Узнаем DNS PTR для нашего IP.

```
maria_rheon@Rheon:~$ nslookup -type=ptr 91.202.45.58
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
58.45.202.91.in-addr.arpa name = ip-45-58.cactus-net.ru.
```

IMAP, SMTP и POP3-серверы Mail.Ru

Для настройки почтовой программы вам потребуется следующая информация:

Электронный адрес	Полное имя почтового ящика, включая логин, @ и домен
Сервер входящей почты (IMAP- и POP3-сервера)	IMAP-сервер — imap.mail.ru POP3-сервер — pop.mail.ru
Сервер исходящей почты (SMTP-сервер)	smtp.mail.ru
Имя пользователя	Полное имя почтового ящика, включая логин, @ и домен
Пароль	Пароль, который вы используете для входа в почтовый ящик
Порт	IMAP — 993 (протокол шифрования SSL/TLS) POP3 — 995 (протокол шифрования SSL/TLS) SMTP — 465 (протокол шифрования SSL/TLS)
Аутентификация	Обычный пароль (без шифрования)

Рисунок 10.1. Информация для подключения

,

Заметим, что в качестве протокола шифрования используются SSl/TLS. Они обеспечивают защищенную передачу данных. Чтобы обеспечить такое подключение мы не можем использовать telnet, так как он работает только с не зашифрованными соединениями. Поэтому воспользуемся следующей командой:

```
maria_rheon@Rheon:~$ gnutls-cli smtp.mail.ru -p 465
Processed 133 CA certificate(s).
Resolving 'smtp.mail.ru:465'...
Connecting to '217.69.139.160:465'...
- Certificate type: X.509
- Got a certificate list of 2 certificates.
- Certificate[0] info:
 - subject `CN=*.mail.ru,OU=IT,O=LLC Mail.Ru,L=Moscow,C=RU',␣issuer␣`CN=GeoTrust
    ↪ ␣RSA␣CA␣2018,OU=www.digicert.com,O=DigiCert␣Inc,C=US', serial 0
    ↪ x0f45c3e7a2b173b7e56b58a6efaaa03c, RSA key 2048 bits, signed using RSA-
    ↪ SHA256, activated `2017-12-15 00:00:00 UTC',␣expires␣`2020-12-14␣12:00:00␣
    ↪ UTC', pin-sha256="FQe4cvpd9CiSqyTOPMEOgm+NzXzth1qQeUCeFsx8AFc="
   Public Key ID:
      sha1:0ca28f9da4ad384258ba65ef9db6b629cb99dd1a
      sha256:1507b872fa5df42892ab24ce3cc10e826f8dcd7ced875a9079409e16cc7c0057
   Public Key PIN:
      pin-sha256:FQe4cvpd9CiSqyTOPMEOgm+NzXzth1qQeUCeFsx8AFc=
   Public key's␣random␣art:
␣␣␣␣+--[␣RSA␣2048]----+
```

```
16 ␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣|
17 ␣␣␣␣|␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣|
18 ␣␣␣␣|␣␣␣␣.␣.␣␣␣␣␣␣␣␣|
19 ␣␣␣␣|␣␣..␣.␣o␣␣␣␣␣␣␣|
20 ␣␣␣␣|.o.␣␣.␣␣␣S␣␣␣␣␣␣␣|
21 ␣␣␣␣|o.oB␣.␣␣␣␣␣␣␣␣␣␣|
22 ␣␣␣␣|.+o.=␣E␣␣␣␣␣␣␣␣␣|
23 ␣␣␣␣|+.␣oo=o=␣␣␣␣␣␣␣␣␣|
24 ␣␣␣␣|o...*=O+.␣␣␣␣␣␣␣␣|
25 ␣␣␣␣+————————————————+
26
27 -␣Certificate[1]␣info:
28 ␣-␣subject␣'CN=GeoTrust␣RSA␣CA␣2018,OU=www.digicert.com,O=DigiCert␣Inc,C=US',
       ↪ issuer 'CN=DigiCert Global Root CA,OU=www.digicert.com,O=DigiCert Inc,C=US
       ↪ ',␣serial␣0x0546fe1823f7e1941da39fce14c46173,␣RSA␣key␣2048␣bits,␣signed␣
       ↪ using␣RSA-SHA256,␣activated␣'2017-11-06␣12:23:45␣UTC', expires '2027-11-06
       ↪  12:23:45 UTC',␣pin-sha256="zUIraRNo+4JoAYA7ROeWjARtIoN4rIEbCpfCRQT6N6A="
29 -␣Status:␣The␣certificate␣is␣trusted.
30 -␣Description:␣(TLS1.2)-(ECDHE-RSA-SECP256R1)-(AES-128-GCM)
31 -␣Session␣ID:␣AD:C3:15:D8:DB:71:4F:04:75:64:4D:5D:76:52:4E:4F:E3:B2:74:96:9A
       ↪ :71:68:3C:3D:E4:3E:5C:F1:C4:63:69
32 -␣Ephemeral␣EC␣Diffie-Hellman␣parameters
33 ␣-␣Using␣curve:␣SECP256R1
34 ␣-␣Curve␣size:␣256␣bits
35 -␣Version:␣TLS1.2
36 -␣Key␣Exchange:␣ECDHE-RSA
37 -␣Server␣Signature:␣RSA-SHA256
38 -␣Cipher:␣AES-128-GCM
39 -␣MAC:␣AEAD
40 -␣Compression:␣NULL
41 -␣Options:␣safe␣renegotiation,
42 -␣Handshake␣was␣completed
43
44 -␣Simple␣Client␣Mode:
45
46 220␣smtp17.mail.ru␣ESMTP␣ready␣(Looking␣for␣Mail␣for␣your␣domain?␣Visit␣https://
       ↪ biz.mail.ru)
```

Первая команда, которую мы должны передать почтовому серверу, - это EHLO или HELO. Это базовое приветствие, которое запускает связь между клиентом и SMTP-сервером. В ответ получим набор команд, которые поддерживаются сервером.

```
1 EHLO ip-45-58.cactus-net.ru.
2 250-smtp17.mail.ru
3 250-SIZE 73400320
4 250-8BITMIME
5 250-PIPELINING
6 250 AUTH PLAIN LOGIN XOAUTH2
```

Пройдем аутентификацию, заранее закодируя логин и пароль через BASE64.

```
1 AUTH LOGIN
2 334 VXNlcm5hbWU6
3 bWlraHNwYnN0dUBtYWlsLnJ1
4 334 UGFzc3dvcmQ6
5 MT....Vx
6 235 Authentication succeeded
```

Следующая команда, которую нужно выполнить, это команда MAIL FROM. Она определяет адрес, с которого будет отправлено письмо. Теперь, когда команда MAIL FROM отправлена, мы можем отправить команду RCPT TO. Эта команда сообщает почтовому

серверу SMTP, кому должно быть отправлено сообщение. Чтобы ввести само сообщение воспользуемся командой DATA. Тема сообщения вводится после слова subject:<>. Признаком окончания сообщения является точка на отдельной строке.

```
MAIL FROM: mikhspbstu@mail.ru
250 OK
RCPT TO: mikhspbstu@mail.ru
250 Accepted
data
354 Enter message, ending with "." on a line by itself
subject: testing
hello from the other side
.
250 OK id=1ga4oo-00011a-UL
```

Сообщение отправлено.

10.4. Подключение к POP3-серверу, проверка почты и получение письма

Теперь попробуем проверить почту и прочитать наше сообщение. Аналогично SMTP:

```
maria_rheon@Rheon:~$ gnutls-cli pop.mail.ru -p 995
Processed 133 CA certificate(s).
Resolving 'pop.mail.ru:995'...
Connecting to '217.69.139.74:995'...
- Certificate type: X.509
- Got a certificate list of 2 certificates.
- Certificate[0] info:
 - subject `CN=*.mail.ru,OU=IT,O=LLC Mail.Ru,L=Moscow,C=RU', issuer `CN=GeoTrust RSA CA 2018,OU=www.digicert.com,O=DigiCert Inc,C=US', serial 0x0f45c3e7a2b173b7e56b58a6efaaa03c, RSA key 2048 bits, signed using RSA-SHA256, activated `2017-12-15 00:00:00 UTC', expires `2020-12-14 12:00:00 UTC', pin-sha256="FQe4cvpd9CiSqyTOPMEOgm+NzXzth1qQeUCeFsx8AFc="
  Public Key ID:
    sha1:0ca28f9da4ad384258ba65ef9db6b629cb99dd1a
    sha256:1507b872fa5df42892ab24ce3cc10e826f8dcd7ced875a9079409e16cc7c0057
  Public Key PIN:
    pin-sha256:FQe4cvpd9CiSqyTOPMEOgm+NzXzth1qQeUCeFsx8AFc=
  Public key's random art:
    +--[ RSA 2048]----+
    |                 |
    |                 |
    |    . .          |
    |  . . . o        |
    |.o. .   S        |
    |o.oB .           |
    |.+o.= E          |
    |+. oo=o=         |
    |o...*=O+.        |
    +-----------------+

- Certificate[1] info:
 - subject `CN=GeoTrust RSA CA 2018,OU=www.digicert.com,O=DigiCert Inc,C=US', issuer `CN=DigiCert Global Root CA,OU=www.digicert.com,O=DigiCert Inc,C=US', serial 0x0546fe1823f7e1941da39fce14c46173, RSA key 2048 bits, signed using RSA-SHA256, activated `2017-11-06 12:23:45 UTC', expires `2027-11-06 12:23:45 UTC', pin-sha256="zUIraRNo+4JoAYA7ROeWjARtIoN4rIEbCpfCRQT6N6A="
- Status: The certificate is trusted.
- Description: (TLS1.2)-(ECDHE-RSA-SECP256R1)-(AES-128-GCM)
```

```
- Session ID: C8:E9:2E:91:83:38:62:7C:70:FF:93:EF:39:FB:3B:D7:8A:94:3C:8F
    ↪ :68:64:76:3A:DA:FD:DE:79:65:DC:84:30
- Ephemeral EC Diffie-Hellman parameters
 - Using curve: SECP256R1
 - Curve size: 256 bits
- Version: TLS1.2
- Key Exchange: ECDHE-RSA
- Server Signature: RSA-SHA256
- Cipher: AES-128-GCM
- MAC: AEAD
- Compression: NULL
- Options: safe renegotiation ,
- Handshake was completed

- Simple Client Mode:
```

Авторизируемся.

```
+OK
USER mikhspbstu@mail.ru
+OK
PASS 12345q
+OK Welcome!
```

С помощью команды STAT получим количество сообщений и общее количество символов в них.

```
STAT
+OK 4706 458762139
```

Запросим список всех сообщений, для каждого из которых будет указан свой размер в символах.

```
LIST
+OK 4706 messages (458762139 octets)
1 30343
2 22184
3 30325
.......
4704 88736
4705 121778
4706 1630
.
```

Прочтем самое новое сообщение. Его id будет последним, так как при получении сообщения оно отправляется в конец списка.

```
RETR 4706
+OK 1630 octets
Delivered-To: mikhspbstu@mail.ru
Return-path: <mikhspbstu@mail.ru>
Received: by smtp17.mail.ru with esmtpa (envelope-from <mikhspbstu@mail.ru>)
  id 1ga4oo-00011a-UL
  for mikhspbstu@mail.ru; Thu, 20 Dec 2018 23:21:37 +0300
subject: testing
Message-Id: <E1ga4oo-00011a-UL.mikhspbstu-mail-ru@smtp17.mail.ru>
From: mikhspbstu@mail.ru
Date: Thu, 20 Dec 2018 23:21:36 +0300
Authentication-Results: smtp17.mail.ru; auth=pass smtp.auth=mikhspbstu@mail.ru
    ↪ smtp.mailfrom=mikhspbstu@mail.ru
X-77F55803: 0014004
    ↪ E1F3277295A78504BD2AC29419868B496DAF47A9F48671ADAC6BE9535ED845B3A5B17128C94BA50C09E8
    ↪
```

```
14 X-7FA49CB5: 0
     ↪ D63561A33F958A552AED9435386BE96724DA5E6BF11EC0D52FEEA0A6DDA46DE8941B15DA834481FA182
     ↪
15 X-Mailru-Sender:
     ↪ C8A6F306A889CEAC8CED9B47B8D6BDB29F3D90BEA54F53FBE02B6BDA663CBDAB7D75C949902A79D2348
     ↪
16 X-Mras: OK
17 X-Mailru-Intl-Transport: d,b26656f
18
19 hello from the other side
20 .
```

По теме сообщения и его содержимому убедились, что сообщения было отправлено корректно.

11. Выводы

В данной работе было реализовано клиент-серверное приложение калькулятора. Данная система обеспечивает параллельную работу нескольких клиентов.
В нашем задании требовалось, реализовать "быстрые"и "долгие"математические операции. При этом в ходе работы наименьшей проблемой стала реализация функций вычислений и передачи данных. Одной из более серьезных проблем была задача принудительного отключения клиентов сервером как в TCP так и в UDP.
В TCP было решено ввести специальные идентификаторы, которые бы передавались в структуре серверу при запуске клиента, а в UDP, где фактически соединение не устанавливается, потребовалось давать каждому клиенту свой порт, по которому впоследствии и происходило бы отключение.
Более того, в UDP нужно было найти новый способ реализации поддержки долгих вычислений. Один из способов заключается в разделении клиента на два потока, один получает, другой отправляет информацию. Но и тут появляется небольшой недочет с рассинхроизацией.
Дальнейшая траектория разработки заключается в возможности работы клиентов на разных IP адресах, но на одинаковых портах. Помимо этого, можно было бы исправить обработку длительных операций.
На основе данной работы мы изучили основные принципы работы протоколов транспортного уровня - TCP и UDP. Сравнение Разница между протоколами TCP и UDP — в так называемой “гарантии доставки”. TCP требует отклика от клиента, которому доставлен пакет данных, подтверждения доставки, и для этого ему необходимо установленное заранее соединение. Также протокол TCP считается надежным, тогда как UDP получил даже именование “протокол ненадежных датаграмм. TCP исключает потери данных, дублирование и перемешивание пакетов, задержки. UDP все это допускает, и соединение для работы ему не требуется. Процессы, которым данные передаются по UDP, должны обходиться полученным, даже и с потерями. TCP контролирует загруженность соединения, UDP не контролирует ничего, кроме целостности полученных датаграмм.