Национальный исследовательский Университет ИТМО
Мегафакультет информационных и трансляционных технологий
Факультет инфокоммуникационных технологий

Инфокоммуникационные системы и технологии

Курсовой проект

Работу выполнил:
Д. С. Зубахин
Группа: К3121
Преподаватель:
Г. А. Карапетян

Санкт-Петербург
2021

Содержание

1. Первая лабораторная работа

1.1. Постановка задачи

1)Провести **анализ сетевой конфигурации** ОС (сетевые интерфейсы, назначение имеющихся сетевых интерфейсов, параметры имеющихся сетевых интерфейсов).
2)Найти назначение команды **ping** используя команду man или с помощью сети интернет, найти ключи команды ping, протестировать несколько найденных ключей, тестируя ключи провести анализ возможностей.
3)Установить базу данных **MariaDB**, настроить её, установить её в автозагрузку.
4)Установить **Apache**, настроить и добавить его в автозагрузку и проверить его работу.
5)Установить, настроить и запустить **FTP-сервер**.
6)Создать пользователя, обеспечить подключение к нему через **SSH** через логин и пароль.

1.2. Реализация

1.2.1. Ход выполнения задачи №1

Выведем список сетевых интерфейсов, выполнив команду ifconfig:

```
dmitry@dmitry-Surface-Pro-6:~$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0×10<host>
        loop txqueuelen 1000 (Локальная петля (Loopback))
        RX packets 2128 bytes 210473 (210.4 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 2128 bytes 210473 (210.4 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.1.4 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::d877:403a:e0f9:1a30 prefixlen 64 scopeid 0×20<link>
        ether b8:31:b5:97:4f:68 txqueuelen 1000 (Ethernet)
        RX packets 26063 bytes 28761533 (28.7 MB)
        RX errors 0 dropped 1201 overruns 0 frame 0
        TX packets 21570 bytes 2771540 (2.7 MB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Листинг 1: Информация о сетевых интерфейсах

Исследуем два сетевых интерфейса:

lo:

Интерфейс lo - это локальная петля, которая имеет IP-адрес 127.0.0.1 и предназначена для сетевого доступа к своему же компьютеру. Любые сообщения, посылаемые на этот канал, немедленно принимаются тем же самым каналом. Любые сообщения, которые отправляются с этого интерфейса, но у которых адрес не Loopback Interface, отбрасываются.
1)P-адрес (ipv4) - 127.0.0.1 он фиксированный и изменению не подлежит.
2)Маска подсети — 255.0.0.0

3) IP-адрес (ipv6) - ::1
4) MTU(Максимальный размер полезного блока данных одного пакета , который может быть передан протоколом без фрагментации) равен 65536 байт.
5)Информация о числе полученных, отправленных, потерянных пакетах (а также их объеме), ошибок и т. д.

wlp1s0:

Интерфейс wlp1s0 является беспроводным и обладает следующими характеристиками:
1) Максимальный размер полезного блока данных одного пакета, который может быть передан протоколом без фрагментации(MTU) равен 1500 байт.
2) Включён (UP), принимает широковещательные пакеты (BROADCAST), принимает групповые пакеты (MULTICAST).
3) IP-адрес (ipv4) - 192.168.1.4
4) Маска подсети - 255.255.255.0
5) IP-адрес (ipv6) - fe80::d877:403a:e0f9:1a30
6) Широковещательный адрес - 192.168.1.255
7) Информация о числе полученных, отправленных, потерянных пакетах (а также их объеме), ошибок и т.д.

1.2.2. Ход выполнения задачи №2

Найдем в консоли назначение команды ping с помощью команды man:

Из информации полученной командой **man** следует, что команда **ping** передает небольшой пакет с данными ICMP и ожидает получить обратно пакет ответа, если получает, то считается что удаленный узел доступен. ICMP или Internet Control Message Protocol(надстройка над протоколом IP, которая используется для передачи служебных сообщений и сообщений об ошибках) может передавать только два типа пакетов - это сообщения с отчетами и сообщения запросов. В свою очередь, сообщения запросов делятся на эхо-запрос и эхо-ответ.

Найдем в консоли назначение ключей команды ping с помощью команды man:

-4 — Использует только IPv4(по умолчанию).
-6 — Использует только IPv6.
-a — Сигнализирует о возвращении пакета с помощью динамика.
-A — В чрезвычайно плохих сетевых условиях мог выйти новый запрос, прежде чем последний возвращен. Новый запрос звона выходит, как только последний ответ получен.
-b — Разрешает отправлять пакеты на широковещательный адрес.
-c (count) — Останавливается после отправки указанного в аргументе кол-ва пакетов.

```
dmitry@dmitry-Surface-Pro-6:~$ ping -c 3 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=8.46 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=5.78 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=64 time=7.33 ms
```

Листинг 2: Результат работы команды **ping** с ключом **-c**

-D — выводить время в виде UNIX timestamp.

```
dmitry@dmitry-Surface-Pro-6:~$ ping -D 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
[1603651247.220396] 64 bytes from 192.168.1.1: icmp_seq=1 ttl=64
↪  time=8.46 ms
[1603651248.219290] 64 bytes from 192.168.1.1: icmp_seq=2 ttl=64
↪  time=5.78 ms
[1603651249.222898] 64 bytes from 192.168.1.1: icmp_seq=3 ttl=64
↪  time=7.33 ms
[1603651250.223463] 64 bytes from 192.168.1.1: icmp_seq=3 ttl=64
↪  time=6.33 ms
[1603651251.226390] 64 bytes from 192.168.1.1: icmp_seq=3 ttl=64
↪  time=7.87 ms
^C
- - - 192.168.1.1 ping statistics - - -
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 5.330/6.961/8.463/1.077 ms
```

Листинг 3: Результат работы команды **ping** с ключом **-D**

Unix Timestamp – это метка времени, которая представляет собой последовательность символов, отражающих количество секунд, прошедших с 1 января 1970 года.

-f — Отправка пакетов без задержки. При отправке пакета печатается ".". Если приходит ответ, то "." удаляется. Может использоваться только суперпользователем.

-h — Справка.

-i — Выдерживает интервал между отправкой пакетов, задаваемый в аргументе. Только суперпользователь может указать интервал меньше 0.2.

-w(deadline) — Указывается максимальное время ожидания возвращения пакета(миллисекунды).

-M — Если нужно найти самый подходящий размер MTU с помощью команды Ping, следует определить начальное значение и уменьшать его до тех пор, пока прекратятся ошибки.

MTU (Maximum Transmission Unit) это максимальный размер пакета, который может быть передан по сети без фрагментации.

-q — Не выводится ничего кроме строки в начале и итоговой информации в конце.

-s — В аргументе задаётся размер передаваемых пакетов.

```
dmitry@dmitry-Surface-Pro-6:~$ ping -s 1000 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 1000(1028) bytes of data.
1008 bytes from 192.168.1.1: icmp_seq=1 ttl=65 time=8.10 ms
1008 bytes from 192.168.1.1: icmp_seq=2 ttl=65 time=7.56 ms
1008 bytes from 192.168.1.1: icmp_seq=3 ttl=65 time=10.5 ms
1008 bytes from 192.168.1.1: icmp_seq=4 ttl=65 time=12.5 ms
1008 bytes from 192.168.1.1: icmp_seq=5 ttl=65 time=11.5 ms
1008 bytes from 192.168.1.1: icmp_seq=6 ttl=65 time=6.75 ms
1008 bytes from 192.168.1.1: icmp_seq=7 ttl=65 time=5.85 ms
1008 bytes from 192.168.1.1: icmp_seq=8 ttl=65 time=2.30 ms
^C
- - - 192.168.1.1 ping statistics - - -
8 packets transmitted, 8 received, 0% packet loss, time 7012ms
rtt min/avg/max/mdev = 2.304/8.125/12.516/3.105 ms
```

Листинг 4: Результат работы команды **ping** с ключом **-s**

1.2.3. Ход выполнения задачи №3

Установим MariaDB и проверим работает ли база данных:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo systemctl status mariadb
[sudo] пароль для dmitry:
● mariadb.service - MariaDB 10.5.6 database server
     Loaded: loaded (/lib/systemd/system/mariadb.service; enabled;
     ↪  vendor preset: enabled)
    Drop-In: /etc//systemd/system/mariadb.service.d
             └─migrated-from-my.cnf-settings.conf
     Active: active (running) since Sun 2020-10-25 23:20:29 MSK; 17min
     ↪  ago
       Docs: man:mariadb(8)
             https://mariadb.com/kb/en/library/systemd/
   Main PID: 9489 (mariadbd)
     Status: "Talking your SQL requests now..."
      Tasks: 9 (limit: 9392)
     Memory: 75.0M
     CGroup: /system.slice/mariadb.service
             └─9489 /usr/sbin/mariadbd

окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]:
↪  Processing databases
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]:
↪  information_schema
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]: mysql
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]:
↪  performance_schema
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]: Phase
↪  6/7: Checking and upgrading
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]:
↪  Processing databases
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]:
↪  information_schema
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]:
↪  performance_schema
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]: Phase
↪  7/7: Running 'FLUSH PRIVATE'
окт 25 23:20:31 dmitry-Surface-Pro-6 /etc/mysql/debian-start[9511]: OK
```

Листинг 5: Вывод статуса базу данных

Настроем базу данных для обеспечения её безопасности.

Добавим БД в автозагрузку:

```
keyboard-setup.service                  enabled enabled
lm-sensors.service                      enabled enabled
mariadb.service                         enabled enabled
ModemManager.service                    enabled enabled
```

Листинг 6: Результат добавления БД в автозагрузку

1.2.4. Ход выполнения задачи №4

Установим apache2 с помощью следующей команды:

```
sudo apt install apache2
```

Листинг 7: Команда для установки apache2

Выведем список правил брандмауэра, чтобы проверить, нужна ли их настройка:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo ufw status
Состояние: активен

В                  Действие        Из
-                  - - - - - -      - -
Apache              DENY            Anywhere
OpenSSH             ALLOW           Anywhere
Apache (v6)         DENY            Anywhere (v6)
OpenSSH (v6)        ALLOW           Anywhere (v6)
```

Листинг 8: Список правил брэндмауэра

Включим самый ограниченный профиль, который будет позволять входящий трафик:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo ufw allow 'Apache'
Правило обновлено
Правило обновлено (v6)
dmitry@dmitry-Surface-Pro-6:~$ sudo ufw status
Состояние: активен

В                  Действие        Из
-                  - - - - - -      - -
Apache              ALLOW           Anywhere
OpenSSH             ALLOW           Anywhere
Apache (v6)         ALLOW           Anywhere (v6)
OpenSSH (v6)        ALLOW           Anywhere (v6)
```

Листинг 9: Обновление правил брэндмауэра

Проверим, находится ли Apache2 в автозагрузке:

```
dmitry@dmitry-Surface-Pro-6:~$ systemctl list-unit-files --type=service
↪  --state=enabled
UNIT FILE                          STATE    VENDOR   PRESET
accounts-daemon.service            enabled  enabled
anacron.service                    enabled  enabled
apache2.service                    enabled  enabled
apparmor.service                   enabled  enabled
autovt@.service                    enabled  enabled
```

Листинг 10: Результат вывода процессов, которые находятся в автозагрузке

Запустим наш сервер и проверим, работает ли он:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo systemctl start apache2
dmitry@dmitry-Surface-Pro-6:~$ sudo systemctl status apache2
[sudo] пароль для dmitry:
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled;
     ↪ vendor preset: enabled)
     Active: active (running) since Mon 2020-10-26 14:11:52 MSK; 27min
     ↪ ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 7405 (apache2)
      Tasks: 55 (limit: 9392)
     Memory: 6.5M
     CGroup: /system.slice/apache2.service
             ├─7405 /usr/sbin/apache2 -k start
             ├─7406 /usr/sbin/apache2 -k start
             └─7407 /usr/sbin/apache2 -k start

окт 26 14:11:52 dmitry-Surface-Pro-6 systemd[1]: Starting The Apache HTTP
↪ Server...
окт 26 14:11:52 dmitry-Surface-Pro-6 apachectl[7404]: AH00558: apache2:
↪ Could not reliably detect
окт 26 14:11:52 dmitry-Surface-Pro-6 systemd[1]: Started The Apache HTTP
↪ Server.
```

Листинг 11: Вывод статуса сервера Apache2

1.2.5. Ход выполнения задачи №5

Установим FTP-server следующей командой:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo apt install vsftpd
```

Листинг 12: Команда, введеная в консоли

Откроем порты 20 и 21 для нормальной работы FTP сервера:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo ufw allow 20/tcp
[sudo] пароль для dmitry:
Правило обновлено
Правило обновлено (v6)
dmitry@dmitry-Surface-Pro-6:~$ sudo ufw allow 21/tcp
Правило обновлено
Правило обновлено (v6)
dmitry@dmitry-Surface-Pro-6:~$ sudo ufw status
Состояние: активен

В                  Действие         Из
-                  - - - - - -      - -
Apache              ALLOW           Anywhere
OpenSSH             ALLOW           Anywhere
20/tcp              ALLOW           Anywhere
21/tcp              ALLOW           Anywhere
Apache (v6)         ALLOW           Anywhere (v6)
OpenSSH (v6)        ALLOW           Anywhere (v6)
20/tcp (v6)         ALLOW           Anywhere (v6)
21/tcp (v6)         ALLOW           Anywhere (v6)
```

Листинг 13: Команды, введеные в консоли

Включим FTP сервер и добавим его в автозагрузку:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo systemctl start vsftpd
dmitry@dmitry-Surface-Pro-6:~$ sudo systemctl enable vsftpd
```

Листинг 14: Команды, введеные в консоли

Проверим состояние FTP сервера командой sudo systemctl status vsftpd:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo systemctl status vsftpd
[sudo] пароль для dmitry:
● vsftpd.service - vsftpd FTP server
        Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled;
        ↪  vendor preset: enabled)
        Active: active (running) since Mon 2020-10-26 18:39:39 MSK; 2min
        ↪  2s ago
      Main PID: 18137 (vsftpd)
         Tasks: 1 (limit: 9392)
        Memory: 660.0K
        CGroup: /system.slice/vsftpd.service
                └─18137 /usr/sbin/vsftpd /etc/vsftpd.conf

окт 26 14:11:52 dmitry-Surface-Pro-6 systemd[1]: Starting vsftpd FTP
↪  Server...
окт 26 14:11:52 dmitry-Surface-Pro-6 systemd[1]: Started vsftpd FTP
↪  Server.
```

Листинг 15: Статус FTP сервера

1.2.6. Ход выполнения задачи №6

Создадим пользователя и установим ему пароль.

Подключимся к пользователю:

```
dmitry@dmitry-Surface-Pro-6:~$ ssh dima@localhost
dima@localhost's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-47-generic x86_64)

 * Documentation:   https://help.ubuntu.com
 * Management:      https://landscape.canonical.com
 * Support:         https://ubuntu.com/advantage

8 обновлений могут быть установлены прямо сейчас.
8 из этих обновлений, являются обновлениями безопасности.
Чтобы просмотреть дополнительные обновления выполните: apt list
 ↪  --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Mon Oct 26 22:57:31 2020 from 127.0.0.1
$
```

Листинг 16: Результат подключения

1.3. Вывод

Мы научились выводить информацию о **сетевых интерфейсах**, исследовать их с помощью, полученной с помощью команды **ifconfig**, информации и узнали назначение сетевых интерфейсов, поднятых на ПК.
Мы научились работать с командой **man** и разобрались в назначении и принципе работы команды **ping**. Мы разобрались в работе нескольких ключей команды **ping** и прочитали как работают остальные. Теперь нам понятен функционал и способы использования команды **ping**.
Мы научились устанавливать, настраивать и подключать базу данных **MariaDB**, а также научились выводить список программ, находящихся в автозагрузке.
Мы научились устанавливать сервер **Apache2**, работать с правилами брандмауэра, запускать веб-сервер и проверять его работоспособность.
Мы научились устанавливать, настраивать и запускать **FTP сервер**.
Мы научились создавать пользователей в системе и обеспечивать подключение к ним через **SSH** с использованием логина и пароля.

2. Вторая лабораторная работа

2.1. Постновка задачи

С помощью доступных материалов разобраться в технологии **Ethernet**. Найти и описать **стандарты Ethernet**. Описать **Ethernet-фреймы**. Установить утилиту **Wireshark**. Проанализировать с ее помощью Ethernet заголовок одного из сгенерированных командой ping пакетов.

2.2. Реализация

2.2.1. Теоретическая информация[1][6][7]

Что такое Ethernet? Ethernet — это набор описаний способов физической передачи сигналов (первоначально по коаксиальному кабелю) на первом уровне модели OSI (физический) и формирования кадров (фреймов) на втором уровне модели OSI (канальный) внутри локальных сетей LAN.

Какие стандарты Ethernet существуют и какие у них характеристики? (Число 10 обозначает номинальную битовую скорость передачи данных стандарта, то есть 10Мбит/с а слово «Base» - метод передачи на одной базовой частоте. Последний символ обозначает тип кабеля.)

- 10Base-5 - коаксиальный кабель диаметром 0,5 дюйма (1дм=2,54см), называемый «толстым» коаксиальным кабелем, с волновым сопротивлением 50Ом. используется как моноканал для всех станций, максимальная длина сегмента 500м. Станция подключаться к кабелю через приемопередатчик - трансивер. Трансивер соединяется с сетевым адаптером разъема DB-15 интерфейсным кабелем AUI. Требуется наличие терминаторов на каждом конце, для поглощения распространяющихся по кабелю сигналов.

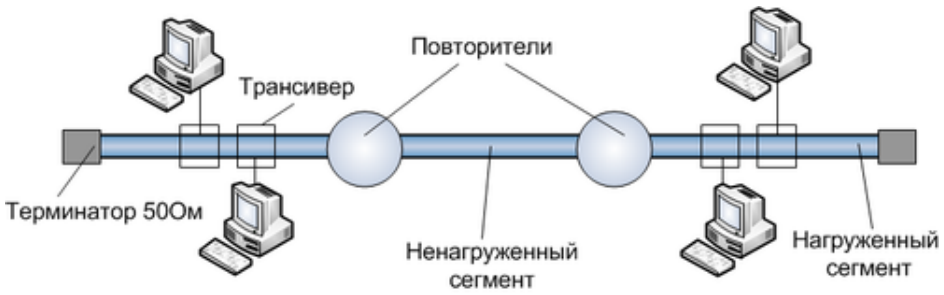

Рисунок 2.1. Стандарт 10Base-5

- 10Base-2 - коаксиальный кабель диаметром 0,25 дюйма, называемый «тонким» коаксиальным кабелем, с волновым сопротивлением 50Ом. Используется как моноканал для всех станций, максимальная длина сегмента 185 м. Для подключения кабеля к сетевой карте нужен Т-коннектор, а на кабеле должен быть BNC-коннектор.

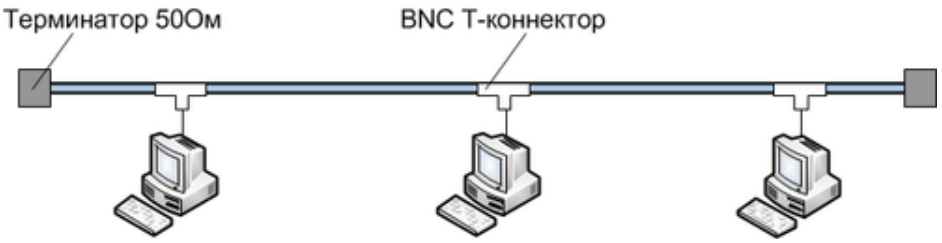

Рисунок 2.2. Стандарт 10Base-2

(Стандарт сетей на коаксиальном кабеле разрешает использование в сети не более 4 повторителей и, соответственно, не более 5 сегментов кабеля. При максимальной длине сегмента кабеля в 500 м это дает максимальную длину сети в 500*5=2500 м. Только 3 сегмента из 5 могут быть нагруженными, то есть такими, к которым подключаются конечные узлы. Между нагруженными сегментами должны быть ненагруженные сегменты.)

- 10Base-T - кабель на основе неэкранированной витой пары (Unshielded Twisted Pair, UTP). Образует звездообразную топологию на основе концентратора, концентратор осуществляет функцию повторителя и образует единый моноканал, максимальная длина сегмента 100м. Конечные узлы соединяются с помощью двух витых пар. Одна пара для передачи данных от узла к концентратору - Tx, а другая для передачи данных от концентратора к узлу – Rx.

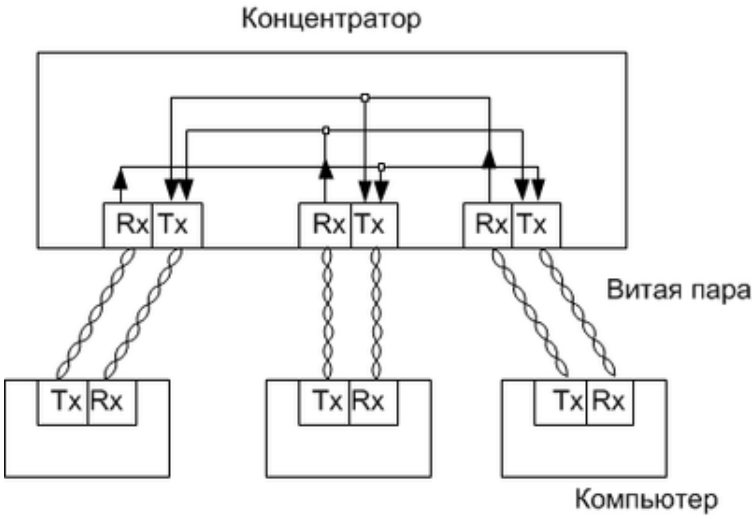

Рисунок 2.3. Стандарт 10Base-T

(В стандарте сетей на витой паре определено максимально число концентраторов между любыми двумя станциями сети, а именно 4.)

- 10Base-F - волоконно-оптический кабель. Функционально сеть Ethernet на оптическом кабеле состоит из тех же элементов, что и сеть стандарта 10Base- T. Стандарт FOIRL (Fiber Optic Inter-Repeater Link) первый стандарт комитета 802.3 для использования оптоволокна в сетях Ethernet. Max длина сегмента 1000м, мах число хабов 4, при общей длине сети не более 2500 м.

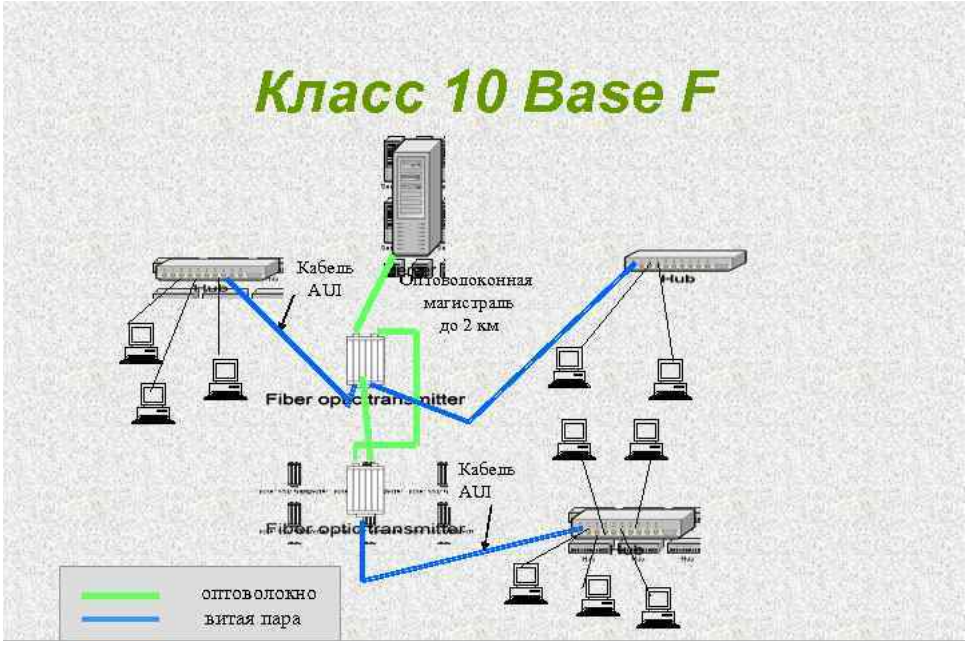

Рисунок 2.4. Стандарт 10Base-F

Какие существуют Ethernet-фреймы и какие у них характеристики?

- **Xerox Ethernet**
  Технология, основанная на коаксиальном кабеле с максимальной скоростью 3 мегабита в секунду. Модификация StarLan, в которой впервые была применена витая пара. Скорость такого соединения невелика – всего 1 мегабит в секунду.

- **Ethernet II (Ethernet DIX)**
  Фирменный стандарт Ethernet компании Xerox, Intel, DEC. Все компьютеры сети подключались к общему коаксиальному кабелю. Коаксиальный кабель (coaxial, от co — совместно и axis — ось, то есть «соосный») – это кабель из пары проводников – центрального провода и окружающего его металлического цилиндра – экрана. Промежуток между проводом и экраном заполнен изоляцией, снаружи кабель так же покрыт изолирующей оболочкой. Такой кабель используется, например, в телевизионных антеннах.

- **IEEE 802.3**
  Юридический стандарт Ethernet.

(Ethernet II и IEEE 802.3 незначительно отличаются. Первый из них исторически раньше появился и при появлении второго много оборудования было на Ethernet II. Сейчас поддерживаются оба. Различие в том, что в Ethernet II передавался тип протокола, а по IEEE 802.3 вместо него передавалась длина поля данных.)

2.2.2. Ход выполнения работы

Установим утилиту Wireshark с помощью следующих команд:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo add-apt-repository
 ↪  ppa:wireshark-dev/stable
dmitry@dmitry-Surface-Pro-6:~$ sudo apt update
dmitry@dmitry-Surface-Pro-6:~$ sudo apt install wireshark
```

Листинг 17: Команды, введенные в консоли

Откроем утилиту Wireshark и проанализируем трафик проходящий через Ethernet интерфейс (enxd0374516eb73)

- Сгенерируем трафик с помощью команды ping, отправив 1 пакет на плоский адрес ya.ru:

```
dmitry@dmitry-Surface-Pro-6:~$ ping -c 1 ya.ru
PING ya.ru (87.250.250.242) 56(84) bytes of data.
64 bytes from ya.ru (87.250.250.242): icmp_seq=1 ttl=250 time=13.0
 ↪  ms

- - - ya.ru ping statistics - - -
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 12.987/12.987/12.987/0.000 ms
```

Листинг 18: Результат работы команды **ping**

- Проанализируем Ethernet заголовок пакета, полученного утилитой **Wireshark**:

```
▶ Frame 176: 98 bytes on wire (784 bits), 98 bytes captured (784
 ↪  bits) on interface enxd0374516eb73, id 0
▼ Ethernet II, Src: Tp-LinkT_16:eb:73 (d0:37:45:16:eb:73), Dat:
 ↪  NetcoreT_ee:c9:a8 (64:ee:b7:ee:c9:a8)
      ▼ Destination: NetcoreT_ee:c9:a8 (64:ee:b7:ee:c9:a8)
            Adress: NetcoreT_ee:c9:a8 (64:ee:b7:ee:c9:a8)
            .... ..0. .... .... .... .... = LG bit: Globally
             ↪  unique address (factory default)
            .... ..0. .... .... .... .... = LG bit: Individual
             ↪  address (unicast)
      ▼ Source: Tp_LinkT_16:eb:73 (d0:37:45:16:eb:73)
            Adress: Tp-LinkT_16:eb:73 (d0:37:45:16:eb:73)
            .... ..0. .... .... .... .... = LG bit: Globally
             ↪  unique address (factory default)
            .... ..0. .... .... .... .... = LG bit: Individual
             ↪  address (unicast)
        Type: IPv4 (0×0800)
▶ Internet Protocol Version 4, Src: 192.168.1.15, Dst:
 ↪  87.250.250.242
▶ Internet Control Message Protocol
```

Листинг 19: Характеристики пакета

1)Информация о пункте назначения(Название устрйоства, IPv4 и IPv6 адрес)
2)Информация о источнике(Название устрйоства, IPv4 и IPv6 адрес)
3)Версия протокола IP с помощью которого был отправлен пакет.
4)Информация о вспомогательном сетевом протоколе (ICMP), который использовался при передаче пакета.

2.3. Вывод

Мы изучили **стандарты Ethernet**, а также **Ethernet-фрэймы**. Также установили утилиту **Wireshark**. Научились пользоваться ей. Сгенерировали трафик с помощью команды **ping** и проанализировали **Ethernet заголовок** одного из сгенерированных пакетов.

3. Третья лабораторная работа

3.1. Постновка задачи

С помощью доступных материалов разобраться в **протоколе IP**. Найти и описать **стандарты протокола IP**. Установить утилиту **Wireshark**. Сгенерировать трафик и проанализировать **IP заголовок** одного из сгенерированных пакетов.

3.2. Реализация

3.2.1. Теоретическая информация[3]

Протокол IP объединяет сегменты сети в единую сеть, обеспечивая доставку пакетов данных между любыми узлами сети через произвольное число промежуточных узлов (маршрутизаторов).

Он классифицируется как протокол **сетевого уровня** по сетевой модели OSI.

IP **не гарантирует надёжной доставки пакета до адресата** — в частности, пакеты могут прийти не в том порядке, в котором были отправлены, продублироваться (приходят две копии одного пакета), оказаться повреждёнными (обычно повреждённые пакеты уничтожаются) или не прийти вовсе. Гарантию безошибочной доставки пакетов дают некоторые протоколы более высокого уровня — транспортного уровня сетевой модели OSI, — например, TCP, которые используют IP в качестве транспорта.

IPv4 — четвёртая версия протокола IP. IPv4 использует 32-битные адреса, ограничивающие адресное пространство 4 294 967 296 возможными уникальными адресами. Традиционной формой записи IPv4 адреса является запись в виде четырёх десятичных чисел (от 0 до 255), разделенных точками. Через дробь указывается длина маски подсети.

Для гибкости в назначении адресов сетей и возможности использовать большое число малых и средних сетей адресное пространство было разделено на несколько логических групп и в каждой группе отводилось разное соотношение хостов и подсетей. Эти группы носят названия классов сетей и пронумерованы латинскими буквами: A, B, C, D и E. Деление основывается на старших битах адреса.

Класс A: 0.XXX.XXX.XXX — 127.XXX.XXX.XXX Первый бит адреса равен нулю, таким образом, класс A **занимает половину всего адресного пространства**. Адрес сети занимает 7 бит, адрес узла — 24 бита, следовательно класс A содержит 128 подсетей по 16 777 216 адресов в каждой.

Класс B: 128.0.XXX.XXX — 191.255.XXX.XXX Класс B **занимает четверть всего адресного пространства**. Адрес сети занимает 14 бит, адрес узла — 16, следовательно класс B содержит 16 384 подсетей по 65 536 адресов в каждой.

Класс C: 192.0.0.XXX — 223.255.255.XXX Класс C **занимает 1/8 адресного пространства**. Адрес сети занимает 21 бит, адрес узла — 8 бит, следовательно класс C содержит 2 097 152 сетей по 256 адресов в каждой.

Класс D: 224.XXX.XXX.XXX — 239.XXX.XXX.XXX Класс D **занимает 1/16 адресного пространства. Используется для многоадресной рассылки**.

Класс E: 240.XXX.XXX.XXX — 255.XXX.XXX.XXX. **Такие адреса запрещены. Зарезервировано для использования в будущем.**

IPv6 — шестая версия протокола IP. Иногда утверждается, что новый протокол может обеспечить до 5·1028 адресов на каждого жителя Земли. Такое большое адресное пространство было введено ради иерархичности адресов (это упрощает маршрутизацию). Тем не менее, увеличенное пространство адресов **сделает NAT необязательным**. Классическое применение IPv6 (по сети /64 на абонента; используется только unicast- адресация) обеспечит возможность использования более 300 млн IP-адресов на каждого жителя Земли.

Улучшения IPv6 по сравнению с IPv4:

- В сверхскоростных сетях **возможна поддержка огромных пакетов (джамбограмм)** — до 4 гигабайт;
- Time to Live переименовано в Hop Limit
- Появились метки потоков и классы трафика
- Появилось многоадресное вещание.

3.2.2. Ход выполнения работы

Установим утилиту Wireshark с помощью следующих команд:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo add-apt-repository
↪  ppa:wireshark-dev/stable
dmitry@dmitry-Surface-Pro-6:~$ sudo apt update
dmitry@dmitry-Surface-Pro-6:~$ sudo apt install wireshark
```

Листинг 20: Команды, введенные в консоли

Узнаем, пакеты с какого интерфейса нас интересуют с помощью команды ifconfig:

```
dmitry@dmitry-Surface-Pro-6:~$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
        inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
        ether 02:42:e6:80:f0:5c txqueuelen 0 (Ethernet)
        RX packets 0 bytes 0 (0.0 B)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 0 bytes 0 (0.0 B)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0×10<host>
        loop txqueuelen 1000 (Локальная петля (Loopback))
        RX packets 2128 bytes 210473 (210.4 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 2128 bytes 210473 (210.4 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.1.4 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::d877:403a:e0f9:1a30 prefixlen 64 scopeid 0×20<link>
        ether b8:31:b5:97:4f:68 txqueuelen 1000 (Ethernet)
        RX packets 26063 bytes 28761533 (28.7 MB)
        RX errors 0 dropped 1201 overruns 0 frame 0
        TX packets 21570 bytes 2771540 (2.7 MB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Листинг 21: Результат работы команды **ifconfig**

Сгенерируем трафик в локальной сети с помощью команды ping:

```
dmitry@dmitry-Surface-Pro-6:~$ ping -c 1 192.168.1.3
PING 192.168.1.3 (192.168.1.3) 56(84) bytes of data.
64 bytes from 192.168.1.3: icmp_seq=1 ttl=64 time=144 ms
64 bytes from 192.168.1.3: icmp_seq=1 ttl=64 time=144 ms
64 bytes from 192.168.1.3: icmp_seq=1 ttl=64 time=144 ms

- - - 192.168.1.3 ping statistics - - -
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 8.790/108.071/171.596/71.112 ms
```

Листинг 22: Результат работы команды **ping**

Выберем один из сгенерированных пакетов, проходящих через интерфейс wlp1s0, и распишем его характеристики:

```
▶ Frame 122: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on
↪  interface wlp1s0, id 0
▼ Ethernet II, Src: Apple_44:88:8f (d4:a3:3d:44:88:8f), Dst:
↪  Microsof_97:4f:68 (b8:31:b5:97:4f:68)
        ▼ Destination: Microsof_97:4f:68 (b8:31:b5:97:4f:68)
                Adress: Microsof_97:4f:68 (b8:31:b5:97:4f:68)
                .... ..0. .... .... .... .... = LG bit: Globally unique
                ↪  address (factory default)
                .... ...0 .... .... .... .... = LG bit: Individual
                ↪  address (unicast)
        ▼ Source: Apple_44:88:8f (d4:a3:3d:44:88:8f)
                Address: Apple_44:88:8f (d4:a3:3d:44:88:8f)
                .... ..0. .... .... .... .... = LG bit: Globally unique
                ↪  address (factory default)
                .... ...0 .... .... .... .... = LG bit: Individual
                ↪  address (unicast)
          Type: IPv4 (0×0800)
▼ Internet Protocol Version 4, Src: 192.168.1.3, Dst: 192.168.1.4
        0100 .... = Version: 4
        .... 0101 = Header Length: 20 bytes (5)
        ▶ Differentiated Services Field: 0×00 (DSCP: CS0, ECN: Not-ECT)
        Total Length: 84
        Identification: 0×0000 (0)
        ▶ Flags: 0×4000, Don't fragment
        Fragment offset: 0
        Time to live: 64
        Protocol: ICMP (1)
        Header checksum 0×b751 [validation disabled]
        [Header checksum status: Unverified]
        Source: 192.168.1.3
        Destination: 192.168.1.4
▶ Internet Control Message Protocol

```

Листинг 23: Характеристики пакета

1) Используется протокол IPv4.

2) Wireshark определяет максимальное количество маршрутизаторов на пути следования пакета. Наличие этого параметра не позволяет пакету бесконечно ходить по сети. Каждый маршрутизатор при обработке пакета должен уменьшить значение TTL на единицу. Пакеты, время жизни которых стало равно нулю, уничтожаются, а отправителю посылается сообщение ICMP Time Exceeded. На отправке пакетов с разным временем жизни основана трассировка их пути прохождения (traceroute). Максимальное значение TTL=255. Обычное начальное значение TTL=64 (зависит от ОС). В нашем случае TTL=64.

3) Указано, какой сетевой протокол использует сгенерированный пакет(ICMP).

4) 16-битная контрольная сумма, используемая для проверки целостности заголовка. Каждый хост или маршрутизатор сравнивает контрольную сумму заголовка со значением этого поля и отбрасывает пакет, если они не совпадают. Целостность данных IP не проверяет — она проверяется протоколами более высоких уровней (такими, как TCP или UDP), которые тоже используют контрольные суммы. Поскольку TTL уменьшается на каждом шаге прохождения пакета, сумма тоже должна вычисляться на каждом

шаге. (0xb751)
5) 32-битный адрес отправителя пакета. Может не совпадать с настоящим адресом отправителя из-за NAT. (192.168.1.3)
6) 32-битный адрес получателя пакета. Также может меняться из-за NAT. (192.168.1.4)

3.3. Вывод

Мы разобрались в 4 и 6 версиях **протокола IP**. Мы установили утилиту **Wireshark**. Научились пользоваться ей. Сгенерировали трафик с помощью команды **ping** и проанализировали **IP заголовок** одного из сгенерированных пакетов.

4. Четвертая лабораторная работа

4.1. Постановка задачи

1)С помощью доступных материалов разобраться в **протоколе DHCP**. Проанализировать **DHCP заголовок** одного из пакетов.
2)С помощью доступных материалов разобраться в **протоколе TCP**. Проанализировать **TCP заголовок** одного из пакетов.
3)С помощью доступных материалов разобраться в **протоколе UDP**. Проанализировать **UDP заголовок** одного из пакетов.

4.2. Реализация

4.2.1. Ход выполнения задачи №1

Теоретическая информация:[2]
DHCP(протокол динамической настройки узла) — сетевой протокол, позволяющий компьютерам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети.

Цель DHCP — устранить два основных ограничения, накладывающихся на другие анагологичные протоколы, а именно: отсутствие поддержки динамического назначения IP-адресов и возможность передавать от сервера на станцию-клиент лишь небольшое число параметров конфигурации.

В **роли транспортного протокола** для обмена DHCP-сообщениями выступает **UDP**.

При отправке сообщения с клиента на сервер используется 67-й порт DHCP-сервера. При передаче в обратном направлении - 68-й.

Алгоритм работы:

- Клиент посылает в свою подсеть широковещательное сообщение **DHCPDISCOVER**, в котором могут указываться устраивающие клиента IP-адрес и срок его аренды. В качестве IP-адреса источника указывается 0.0.0.0, в качестве адреса назначения - 255.255.255.255. Если DHCP-сервер отсутствует в подсети, то **сообщение будет передано в другие подсети** агентами протокола BOOTP.
- **Получив запрос от клиента**, DHCP-сервер отвечает на него сообщением **DHCPOFFER**. В сообщение включается предлагаемый IP-адрес (yiaddr) и прочие конфигурации для клиента (адреса маршрутизаторов, DNS-серверов и т.д.). (На данном этапе сервер не обязан резервировать адрес, который он отправил клиенту)
- **Получив конфигурации от серверов** (их может быть несколько, если в подсети более одного DHCP-севрера), клиент отправляет **широковещательное сообщение DHCPREQUEST**. В нем содержатся идентификатор выбранного сервера и, возможно, желательные значения запрашиваемых параметров конфигурации.(На данном этапе допускается, что клиента не устроит ни один из предложенных адресов, тогда он вновь отправит **DHCPDISCOVER**)

- **Получив DHCPREQUEST и убедившись, что в сообщении его идентификатор**, сервер проверяет свободен ли в данный момент запрошенный адрес. Если да, то отправляет **DHCPACK** и вносит запись в базу, иначе отправляет **DHCPNACK**.
- **Получив сообщение DHCPACK**, клиент обязан убедиться в уникальности IP-адреса (средствами протокола ARP) и зафиксировать суммарный срок его аренды.(Время, прошедшее между отправкой сообщения **DHCPREQUEST** и приемом ответного сообщения **DHCPACK** + срок аренды, указанный в **DHCPACK**. Если адрес уже используется другой станцией, клиент отправляет **DHCPDECLINE** и начинает всю процедуру снова.)
- Для досрочного прекращения аренды адреса клиент отправляет серверу сообщение **DHCPRELEASE**.

Заголовок DHCP пакета:

- **op** Тип сообщения (1 = BOOTREQUEST, 2 = BOOTREPLY)
- **htype** Тип адреса оборудования
- **hlen** Длина адреса оборудования
- **hops** Используется ретранслирующим агентом
- **xid** Идентификатор транзакции между сервером и клиентом
- **secs** Время с момента выдачи DHCPREQUEST или начала обновления конфигурации
- **flags** Флаги (первый бит маркирует широковещательные сообщения)
- **ciaddr** IP-адрес клиента
- **yiaddr** <Ваш> (клиентский) IP-адрес
- **siaddr** IP-адрес следующего сервера, участвующего в загрузке
- **giaddr** IP-адрес ретранслирующего агента
- **chaddr** <Аппаратный> адрес клиента
- **sname** Хост-имя сервера (опция)
- **file** Имя загрузочного файла
- **options** Поле дополнительных параметров

Ход выполнения работы:

- Установим утилиту DHCPING с помощью следующей команды:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo apt-get install dhcping
```

Листинг 24: Команда, введёная в консоли

(Утилита DHCPING проверяет DHCP-сервер используя unicast пакеты.)

- Для того чтобы найти IP адрес DHCP-сервера воспользуемся следующей командой:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo dhcping -s 192.168.0.255
```

Листинг 25: Команда, введёная в консоли

- После того как мы получили ответ с IP адреса 192.168.0.1 отправим на него пакет используя следующую команду:

```
dmitry@dmitry-Surface-Pro-6:~$ sudo dhcping -s 192.168.0.1
```

Листинг 26: Команда, введёная в консоли

- Сгенерируем трафик с помощью утилиты dhcping

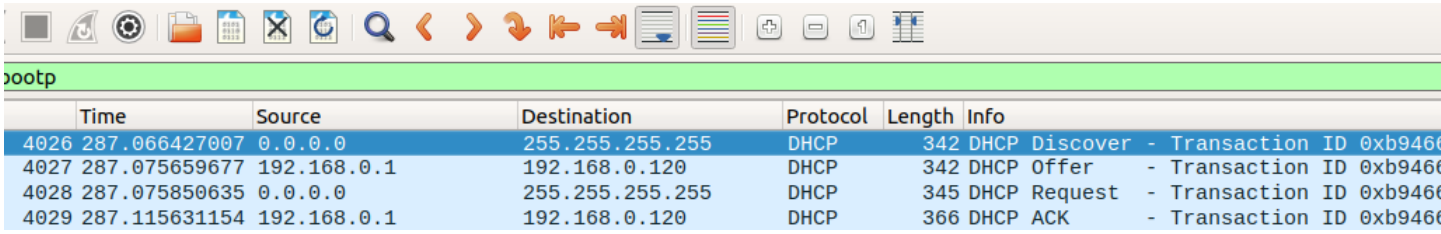

Рисунок 4.1. Сгенерированные пакеты

- Поочерёдно проанализируем четыре пакета:
  - **DHCP Discover**

```
▶ Frame 4026: 342 bytes on wire (2736 bits), 342 bytes captured
  ↪  (2736 bits) on interface wlp1s0
▶ Ethernet II, Src: d4:a3:3d:44:88:8f (d4:a3:3d:44:88:8f), Dst:
  ↪  Broadcast (ff:ff:ff:ff:ff:ff)
▶ Internet Protocol Version 4, Src: 0.0.0.0, Dst:
  ↪  255.255.255.255
▶ User Datagram Protocol, Src Port: 68, Dst Port: 67
▼ Bootstrap Protocol (Discover)
        Message type: Boot Request (1)
        Hardware type: Ethernet (0×01)
        Hardware address length: 6
        Hops: 0
        Transaction ID: 0×b9466634
        Seconds elapsed: 0
      ▶ Bootp flags: 0×0000 (Unicast)
        Client IP address: 0.0.0.0
        Your (client) IP address: 0.0.0.0
        Next server IP address: 0.0.0.0
        Relay agent IP address: 0.0.0.0
        Client MAC address: d4:a3:3d:44:88:8f
         ↪  (d4:a3:3d:44:88:8f)
        Client hardware address padding: 00000000000000000000
        Server host name not given
        Boot file name not given
        Magic cookie: DHCP
▶ Option: (53) DHCP Message Type (Discover)
▶ Option: (12) Host Name
▶ Option: (55) Parameter Request List
▶ Option: (255) End
```

Листинг 27: DHCP(Discover) заголовок пакета

1)Тип сообщения: BOOTREQUEST

2)Тип адреса оборудования: Ethernet

3)Длина адреса оборудования: 6 байт

4)Не используется ретранслирующим агентом

5)Индентификатор транзакции между сервером и клентом: 0xb9466634

6)Время с момента выдачи DHCPREQUEST или начала обновления конфигурации: 0 сек

7)IP адрес клиента: 0.0.0.0

8)"Наш" IP адрес: 0.0.0.0

9)IP-адрес следующего сервера, участвующего в загрузке: 0.0.0.0

10)IP-адрес ретранслирующего агента: 0.0.0.0

11)<Аппаратный> адрес клиента: 00000000000000000000

12)Хост-имя сервера (опция): Не дано

13)Имя загрузочного файла: Не дано

14)Поле дополнительных параметров: Тип сообщений DHCP, Имя хоста, Запрашиваемые параметры

– **DHCP Offer**

```
1  ▶ Frame 4027: 342 bytes on wire (2736 bits), 342 bytes captured
   ↪  (2736 bits) on interface wlp1s0
2  ▶ Ethernet II, Src: d4:a3:3d:44:88:8f (d4:a3:3d:44:88:8f), Dst:
   ↪  c8:58:c0:b5:ae:f0 (c8:58:c0:b5:ae:f0)
3  ▶ Internet Protocol Version 4, Src: 192.168.0.1, Dst:
   ↪  192.168.0.120
4  ▶ User Datagram Protocol, Src Port: 67, Dst Port: 68
5  ▼ Bootstrap Protocol (Offer)
6          Message type: Boot Reply (2)
7          Hardware type: Ethernet (0×01)
8          Hardware address length: 6
9          Hops: 0
10         Transaction ID: 0×b9466634
11         Seconds elapsed: 0
12       ▶ Bootp flags: 0×0000 (Unicast)
13         Client IP address: 0.0.0.0
14         Your (client) IP address: 192.168.0.120
15         Next server IP address: 192.168.0.1
16         Relay agent IP address: 0.0.0.0
17         Client MAC address: c8:58:c0:b5:ae:f0
            ↪  (c8:58:c0:b5:ae:f0)
18         Client hardware address padding: 00000000000000000000
19         Server host name not given
20         Boot file name not given
21         Magic cookie: DHCP
22 ▶ Option: (53) DHCP Message Type (Offer)
23 ▶ Option: (54) DHCP Server Identifier
24 ▶ Option: (51) IP Address Lease Time
25 ▶ Option: (58) Renewal Time Value
26 ▶ Option: (59) Rebinding Time Value
27 ▶ Option: (1) Subnet Mask
28 ▶ Option: (28) Broadcast Address
29 ▶ Option: (6) Domain Name Server
30 ▶ Option: (3) Router
31 ▶ Option: (255) End
```

Листинг 28: DHCP(Offer) заголовок пакета

1)Тип сообщения: BOOTREPLY

2)Тип адреса оборудования: Ethernet

3)Длина адреса оборудования: 6 байт

4)Не используется ретранслирующим агентом

5)Индентификатор транзакции между сервером и клентом: 0xb9466634

6)Время с момента выдачи DHCPREQUEST или начала обновления конфигурации: 0 сек

7)IP адрес клиента: 0.0.0.0

8)"Наш" IP адрес: 192.168.0.120

9)IP-адрес следующего сервера, участвующего в загрузке: 192.168.0.1

10)IP-адрес ретранслирующего агента: 0.0.0.0

11)<Аппаратный> адрес клиента: 00000000000000000000

12)Хост-имя сервера (опция): Не дано

13)Имя загрузочного файла: Не дано

14)Поле дополнительных параметров: Тип сообщений DHCP, Индентификатор DHCP сервера, Время аренды IP адреса, Маска подсети, Широковещательный адрес и тд.

– **DHCP Request**

```
▶ Frame 4028: 345 bytes on wire (2760 bits), 345 bytes captured
 ↪  (2760 bits) on interface wlp1s0
▶ Ethernet II, Src: c8:58:c0:b5:ae:f0 (c8:58:c0:b5:ae:f0), Dst:
 ↪  Broadcast (ff:ff:ff:ff:ff:ff)
▶ Internet Protocol Version 4, Src: 0.0.0.0, Dst:
 ↪  255.255.255.255
▶ User Datagram Protocol, Src Port: 68, Dst Port: 67
▼ Bootstrap Protocol (Request)
       Message type: Boot Request (1)
       Hardware type: Ethernet (0×01)
       Hardware address length: 6
       Hops: 0
       Transaction ID: 0×b9466634
       Seconds elapsed: 0
     ▶ Bootp flags: 0×0000 (Unicast)
       Client IP address: 0.0.0.0
       Your (client) IP address: 0.0.0.0
       Next server IP address: 0.0.0.0
       Relay agent IP address: 0.0.0.0
       Client MAC address: c8:58:c0:b5:ae:f0
        ↪  (c8:58:c0:b5:ae:f0)
       Client hardware address padding: 00000000000000000000
       Server host name not given
       Boot file name not given
       Magic cookie: DHCP
▶ Option: (53) DHCP Message Type (Request)
▶ Option: (54) DHCP Server Identifier
▶ Option: (50) Requested IP Address
▶ Option: (12) Host Name
▶ Option: (55) Parameter Request List
▶ Option: (255) End
```

Листинг 29: DHCP(Request) заголовок пакета

1)Тип сообщения: BOOTREQUEST

2)Тип адреса оборудования: Ethernet

3)Длина адреса оборудования: 6 байт

4)Не используется ретранслирующим агентом

5)Индентификатор транзакции между сервером и клентом: 0xb9466634

6)Время с момента выдачи DHCPREQUEST или начала обновления конфигурации: 0 сек

7)IP адрес клиента: 0.0.0.0

8)"Наш" IP адрес: 0.0.0.0

9)IP-адрес следующего сервера, участвующего в загрузке: 0.0.0.0

10)IP-адрес ретранслирующего агента: 0.0.0.0

11)<Аппаратный> адрес клиента: 00000000000000000000

12)Хост-имя сервера (опция): Не дано

13)Имя загрузочного файла: Не дано

14)Поле дополнительных параметров: Тип сообщений DHCP, Индентификатор DHCP сервера, Занятый IP адрес, Имя хоста, Список запрошенных параметров

– **DHCP ACK**

```
1  ▶ Frame 4029: 366 bytes on wire (2928 bits), 366 bytes captured
   ↪  (2928 bits) on interface wlp1s0
2  ▶ Ethernet II, Src: d4:a3:3d:44:88:8f (d4:a3:3d:44:88:8f), Dst:
   ↪  c8:58:c0:b5:ae:f0 (c8:58:c0:b5:ae:f0)
3  ▶ Internet Protocol Version 4, Src: 192.168.0.1, Dst:
   ↪  192.168.0.120
4  ▶ User Datagram Protocol, Src Port: 67, Dst Port: 68
5  ▼ Bootstrap Protocol (ACK)
6          Message type: Boot Reply (2)
7          Hardware type: Ethernet (0×01)
8          Hardware address length: 6
9          Hops: 0
10         Transaction ID: 0×b9466634
11         Seconds elapsed: 0
12       ▶ Bootp flags: 0×0000 (Unicast)
13         Client IP address: 0.0.0.0
14         Your (client) IP address: 192.168.0.120
15         Next server IP address: 192.168.0.1
16         Relay agent IP address: 0.0.0.0
17         Client MAC address: c8:58:c0:b5:ae:f0
           ↪  (c8:58:c0:b5:ae:f0)
18         Client hardware address padding: 00000000000000000000
19         Server host name not given
20         Boot file name not given
21         Magic cookie: DHCP
22 ▶ Option: (53) DHCP Message Type (ACK)
23 ▶ Option: (54) DHCP Server Identifier
24 ▶ Option: (51) IP Address Lease Time
25 ▶ Option: (58) Renewal Time Value
26 ▶ Option: (59) Rebinding Time Value
27 ▶ Option: (1) Subnet Mask
28 ▶ Option: (28) Broadcast Address
29 ▶ Option: (6) Domain Name Server
30 ▶ Option: (12) Host Name
31 ▶ Option: (3) Router
32 ▶ Option: (255) End
```

Листинг 30: DHCP(ACK) заголовок пакета

1)Тип сообщения: BOOTREPLY

2)Тип адреса оборудования: Ethernet

3)Длина адреса оборудования: 6 байт

4)Не используется ретранслирующим агентом

5)Индентификатор транзакции между сервером и клентом: 0xb9466634

6)Время с момента выдачи DHCPREQUEST или начала обновления конфигурации: 0 сек

7)IP адрес клиента: 0.0.0.0

8)"Наш" IP адрес: 192.168.0.120

9)IP-адрес следующего сервера, участвующего в загрузке: 192.168.0.1

10)IP-адрес ретранслирующего агента: 0.0.0.0

11)<Аппаратный> адрес клиента: 00000000000000000000

12)Хост-имя сервера (опция): Не дано

13)Имя загрузочного файла: Не дано

14)Поле дополнительных параметров: Тип сообщений DHCP, Индентификатор DHCP сервера, Время аренды IP адреса, Маска подсети, Широковещательный адрес, Имя хоста и тд.

4.2.2. Ход выполнения задачи №2

Теоретическая информация:[4]

Transmission Control Protocol — один из основных протоколов передачи данных интернета, предназначенный для **управления передачей данных**. Пакеты в TCP называются **сегментами**.

В стеке протоколов TCP/IP выполняет функции **транспортного уровня** модели OSI.

Механизм TCP предоставляет поток данных с предварительной установкой соединения, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета, гарантируя тем самым, в отличие от UDP, целостность передаваемых данных и уведомление отправителя о результатах передачи.

Заголовок сегмента TCP:

- **Порт источника, Порт назначения**
  Эти 16-битные поля содержат номера портов — числа, которые определяются по специальному списку.

  Порт источника идентифицирует приложение клиента, с которого отправлены пакеты. Ответные данные передаются клиенту на основании этого номера.

  Порт назначения идентифицирует порт, на который отправлен пакет.

- **Порядковый номер**
  Sequence number (32 бита) — измеряется в байтах, и каждый переданный байт полезных данных (payload) увеличивает это значение на 1.

  Если установлен **флаг SYN** (идёт установление сессии), то поле содержит **изначальный порядковый номер — ISN (Initial Sequence Number)**. В целях безопасности это значение генерируется случайным образом и может быть равно от 0 до $2^{32} - 1$ (4294967295). Первый байт полезных данных в устанавливающейся сессии будет иметь номер ISN+1.

  В противном случае, **если SYN не установлен**, первый байт данных, передаваемый в данном пакете, **имеет этот порядковый номер**.

- **Номер подтверждения**
  Acknowledgment Number (ACK SN) (32 бита) — если установлен флаг ACK, то это поле содержит порядковый номер октета, который отправитель данного сегмента желает получить. Это означает, что все предыдущие октеты (с номерами от ISN+1 до ACK-1 включительно) были успешно получены.

  Каждая сторона подсчитывает свой **Sequence number** для переданных данных и отдельно **Acknowledgement number** для полученных данных. **Sequence number** каждой из сторон соответствует **Acknowledgement number** другой стороны.

- **Длина заголовка (смещение данных)**
  Длина заголовка (Data offset) занимает 4 бита и указывает значение длины заголовка, измеренное в 32-битовых словах. Минимальный размер составляет 20 байт (пять 32-битовых слов), а максимальный — 60 байт (пятнадцать 32-битовых слов). Длина заголовка определяет смещение полезных данных относительно начала сегмента.

- **Зарезервировано**
  Зарезервировано (6 бит) для будущего использования и должно устанавливаться в ноль. Из них два (5-й и 6-й) уже определены:

  - **CWR (Congestion Window Reduced)** — Поле «Окно перегрузки уменьшено» — флаг установлен отправителем, чтобы указать, что получен пакет с установленным флагом ECE.
  - **ECE (ECN-Echo)** — Поле «Эхо ECN» — указывает, что данный узел способен на ECN (явное уведомление перегрузки) и для указания отправителю о перегрузках в сети.

- **Флаги (управляющие биты)**
  Это поле содержит 6 битовых флагов:

  - **URG — поле «Указатель важности»** задействовано. Когда узел отправляет сегмент с URG флагом, то узел-получатель принимает его на отдельном канале.
  - **ACK — поле «Номер подтверждения»** задействовано.
  - **PSH** — инструктирует получателя протолкнуть данные, накопившиеся в приёмном буфере, в приложение пользователя. API для установки PSH флага нет. Обычно он устанавливается ядром,когда оно очищает буфер.(Дело в том, что когда узел отправляет информацию, TCP сохраняет ее в буфере и не передает ее сразу другому узлу ,ожидая ,если узел-отправитель захочет передать еще. Такая же схема работает и у узла-получателя. Когда он получает информацию, TCP сохраняет ее в буфере,чтобы не тревожить приложение из-за каждого байта полученной информации.)Если узел отправляет сегмент с PSH флагом, это значит, что он отправил все, что было нужно.
  - **RST** — оборвать соединения, сбросить буфер (очистка буфера) (англ. Reset the connection)
  - **SYN** — синхронизация номеров последовательности (англ. Synchronize sequence numbers)
  - **FIN** — флаг, будучи установлен, указывает на завершение соединения.

- **Размер окна**
  Window Size определяет количество байт данных (payload), после передачи которых отправитель ожидает подтверждения от получателя, что данные получены. Иначе говоря, получатель пакета располагает для приёма данных буфером длиной "размер окна" байт.

  По умолчанию размер окна измеряется в байтах, поэтому ограничен 216 (65535) байтами. Однако благодаря TCP опции **Window scale option** этот размер может быть увеличен до 1 Гбайта. Чтобы задействовать эту опцию, обе стороны должны **согласовать это в своих SYN сегментах**.

- **Контрольная сумма (Checksum)**
  Поле контрольной суммы — это **16-битное дополнение к сумме всех 16-битных слов заголовка (включая псевдозаголовок) и данных**. Если сегмент, по которому вычисляется контрольная сумма, имеет длину не кратную 16-битам, то длина сегмента увеличивается до кратной 16-ти, за счёт дополнения к нему справа нулевых битов заполнения. Биты заполнения (0) не передаются в сообщении и служат только для расчёта контрольной суммы. При расчёте контрольной суммы значение самого поля контрольной суммы принимается равным 0.

- **Указатель важности (Urgent pointer)**
  16-битовое значение положительного смещения от порядкового номера в данном сегменте. Это поле **указывает порядковый номер октета, которым заканчиваются важные (urgent) данные**. Поле принимается во внимание только для пакетов с установленным флагом URG.

- **Опции**
  Могут применяться в некоторых случаях **для расширения протокола**. **Иногда используются для тестирования**. На данный момент в опции практически всегда включают 2 байта NOP (в данном случае 0x01) и 10 байт, задающих timestamps. Вычислить длину поля опции можно через значение поля смещения.

Ход выполнения работы:

```
▶ Frame 4015: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
 ↪  on interface wlp1s0, id 0
▶ Ethernet II, Src: d4:a3:3d:44:88:8f (d4:a3:3d:44:88:8f), Dst:
 ↪  Microsof_97:4f:68 (b8:31:b5:97:4f:68)
▶ Internet Protocol Version 4, Src: 192.168.0.120, Dst: 93.186.225.146
▼ Transmission Control Protocol, Src Port:39786, Dst Port: 443, Seq:
 ↪  10857, Ack:11949, Len: 0
        Source Port: 39786
        Destination Port: 443
        [Stream index: 2]
        [TCP Segment Len: 0]
        Sequence number: 10857    (relative sequence number)
        [Next sequence number: 10857    (relative sequence number)]
        Acknowledgment number: 11949    (relative ack number)
        1000 .... = Header Length: 32 bytes (8)
      ▶ Flags: 0×010 (ACK)
        Window size value: 567
        [Calculated window size: 567]
        [Window size scaling factor: -1 (unknown)]
        Checksum: 0×00c8 [unverified]
        [Checksum Status: Unverified]
        Urgent pointer: 0
      ▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP),
       ↪  Timestamps
      ▶ [SEQ/ACK analysis]
      ▶ [Timestamps]
```

Листинг 31: TCP заголовок сегмента

Анализ TCP заголовка:

1)Порт источника: 39786; Порт назначения: 443

2)Порядковый номер сегмента: 10857

3)Номер подтверждения сегмента: 11949

4)Длина заголовка: 32 байта

5)Размер окна: 567

6)Контрольная сумма: 0x00c8

7)Указатель важности: 0

4.2.3. Ход выполнения задачи №3

Теоретическая информация:[5]

User Datagram Protocol — один из ключевых элементов набора сетевых протоколов для Интернета. С UDP компьютерные приложения могут посылать датаграммы другим хостам по IP-сети без необходимости предварительного сообщения для установки специальных каналов передачи или путей данных.

UDP использует **простую модель передачи, без неявных «рукопожатий» для обеспечения надёжности**, упорядочивания или целостности данных. Таким образом, UDP предоставляет ненадёжный сервис, и датаграммы могут прийти не по порядку, дублироваться или вовсе исчезнуть без следа.

UDP подразумевает, что проверка ошибок и исправление **либо не нужны, либо должны исполняться в приложении**. **Чувствительные ко времени приложения** часто используют UDP, так как предпочтительнее сбросить пакеты, чем ждать задержавшиеся пакеты, что может оказаться невозможным в системах реального времени.

При необходимости исправления ошибок на сетевом уровне интерфейса приложение может **задействовать TCP или SCTP**, разработанные для этой цели.

Природа UDP как протокола без сохранения состояния также **полезна для серверов, отвечающих на небольшие запросы от огромного числа клиентов**, например DNS и потоковые мультимедийные приложения вроде IPTV, Voice over IP, протоколы туннелирования IP и многие онлайн-игры.

Заголовок датаграммы UDP:

Заголовок UDP состоит из четырёх полей, каждое по 2 байта (16 бит). **Два из них необязательны к использованию в IPv4(Порт отправителя и контрольная сумма)**, в то время как в IPv6 необязателен только порт отправителя.

- **Порт отправителя**
  В этом поле указывается **номер порта отправителя**.

Предполагается, что это значение задаёт порт, на который при необходимости **будет посылаться ответ**. В противном же случае, значение должно быть равным 0.

- **Порт получателя**
  Это поле **обязательно** и **содержит порт получателя**.

  Аналогично порту отправителя, если хостом-получателем является клиент, то номер порта динамический, если получатель — сервер, то это будет «хорошо известный» порт.

- **Длина датаграммы**

  Поле, **задающее длину всей датаграммы** (заголовка и данных) в байтах. **Минимальная длина равна** длине заголовка — 8 байт. Теоретически, **максимальный размер поля** — 65535 байт для UDP-датаграммы (8 байт на заголовок и 65527 на данные). Фактический предел для длины данных при использовании IPv4 — 65507 (помимо 8 байт на UDP-заголовок требуется ещё 20 на IP-заголовок).

  В IPv6 пакеты UDP могут иметь больший размер. Максимальное значение составляет 4 294 967 295 байт (2^{32}—1), из которых 8 байт соответствуют заголовку, а остальные 4 294 967 287 байт — данным.

- **Контрольная сумма**

  Поле контрольной суммы используется для **проверки заголовка и данных на ошибки**. Если сумма не сгенерирована передатчиком, то поле заполняется нулями. Поле не является обязательным для IPv4.

Ход выполнения работы:

```
▶ Frame 687: 101 bytes on wire (808 bits), 101 bytes captured (808 bits)
 ↪  on interface wlp1s0
▶ Ethernet II, Src: d4:a3:3d:44:88:8f (d4:a3:3d:44:88:8f), Dst:
 ↪  Microsof_97:4f:68 (b8:31:b5:97:4f:68)
▶ Internet Protocol Version 4, Src: 192.168.0.120, Dst: 192.168.0.120
▼ User Datagram Protocol, Src Port: 53, Dst Port: 40654
        Source Port: 53
        Destination Port: 40654
        Length: 67
        Checksum: 0×f2ae [unverified]
        [Checksum Status: Unverified]
        [Stream index: 29]
▶ Domain Name System (response)
```

Листинг 32: UDP заголовок датаграммы

Анализ UDP заголовка:

1)Порт отправителя: 53; Порт получателя: 40654

2)Длина датаграммы: 67

3)Контрольная сумма: 0xf2ae

4.3. Вывод

С помощью доступных материалов мы разобрались в **протоколе DHCP** и проанализировали **DHCP заголовок** одного из пакетов.
С помощью доступных материалов мы разобрались в **протоколе TCP** и проанализировали **TCP заголовок** одного из пакетов.
С помощью доступных материалов мы разобрались в **протоколе UDP** и проанализировали **UDP заголовок** одного из пакетов.

Список использованных источников

1. Ethernet фрэймы. — URL: https://neerc.ifmo.ru/wiki/index.php?title=Ethernet.
2. Информация о протоколе DHCP. — URL: https://neerc.ifmo.ru/wiki/index.php?title=DHCP.
3. Информация о протоколе IP. — URL: https://neerc.ifmo.ru/wiki/index.php?title=IP.
4. Информация о протоколе TCP. — URL: https://anisim.org/articles/tcp-protokol/#TCP.
5. Информация о протоколе UDP. — URL: https://anisim.org/articles/udp-protokol/.
6. Общая информация. — URL: https://www.insotel.ru/press/articles/stroim_set_ethernet_lvs/tehnologiya_ethernet_obzor_tehnologii_raznovidnosti_ethernet_standarty_ethernet/.
7. Стандарты Ethernet. — URL: https://ru.bmstu.wiki/Ethernet.