

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

КАФЕДРА КОМПЬЮТЕРНЫХ СИСТЕМ И ПРОГРАММНЫХ ТЕХНОЛОГИЙ

**Отчёт по лабораторной работе №1**

**Курс: «Защита информации»**

**Тема: «Исследование сетевого трафика»**

Выполнил студент:

Бояркин Никита Сергеевич

Группа: 43501/3

Проверил:

Новопашенный Андрей Гелиевич

Санкт-Петербург  
2017 г.

# Содержание

<b>1</b>	<b>Лабораторная работа №1</b>	<b>2</b>
1.1	Цель работы . . . . .	2
1.2	Программа работы . . . . .	2
1.3	Конфигурация сети . . . . .	2
1.4	Ход работы . . . . .	3
1.4.1	Утилита Ping . . . . .	3
1.4.2	Утилита Tracert . . . . .	5
1.4.3	Протокол ICMP . . . . .	6
1.4.4	Протокол ARP . . . . .	7
1.4.5	Протокол TCP . . . . .	7
1.5	Вывод . . . . .	11
1.6	Приложение 1 . . . . .	12
1.6.1	Установка соединения с закрытым портом с отключенным межсетевым экраном . . . . .	12

# Лабораторная работа №1

## 1.1 Цель работы

Получение навыков по исследованию сетевого трафика.

## 1.2 Программа работы

При помощи программы Wireshark продемонстрировать сетевой трафик для:

- Утилиты ping
  - Без фрагментации
  - С фрагментацией
- Утилиты tracert
- Протокола ICMP
- Протокола ARP
  - Запрос
  - Ответ
- Протокола TCP
  - Установление соединения
  - Разрыв соединения
  - Попытка соединения на отсутствующий порт

## 1.3 Конфигурация сети

```
Адаптер Ethernet Ethernet:
DNS-суффикс подключения . . . . . :
Описание. . . . . : Intel(R) Ethernet Connection (2) I218-V
Физический адрес. . . . . : 4C-CC-6A-25-CC-59
DHCP включен. . . . . : Да
Автонастройка включена. . . . . : Да
Локальный IPv6-адрес канала . . . : fe80::b5e4:68bb:4d4c:fad7%11(Основной)
IPv4-адрес. . . . . : 192.168.0.106(Основной)
Маска подсети. . . . . : 255.255.255.0
Аренда получена. . . . . : 11 марта 2017 г. 9:59:41
Срок аренды истекает. . . . . : 11 марта 2017 г. 22:59:42
Основной шлюз. . . . . : 192.168.0.1
DHCP-сервер. . . . . : 192.168.0.1
IAID DHCPv6 . . . . . : 38587498
DUID клиента DHCPv6 . . . . . : 00-01-00-01-1F-76-F1-20-4C-CC-6A-25-CC-59
DNS-серверы. . . . . : 192.168.0.1
NetBios через TCP/IP. . . . . : Включен
```

Рис. 1.1: Сетевые параметры компьютера

## 1.4 Ход работы

### 1.4.1 Утилита Ping

Утилита Ping отправляет эхо-запрос ICMP, после чего, в случае успеха должен прийти симметричный эхо-ответ ICMP. Если пакет не пришел за некоторое время, то удаленный сервер считается недостижимым. По умолчанию производится четыре попытки.

#### Ping без фрагментации

Трафик утилиты Ping со стандартными параметрами ( $bytes = 128, TTL = 128$ ):

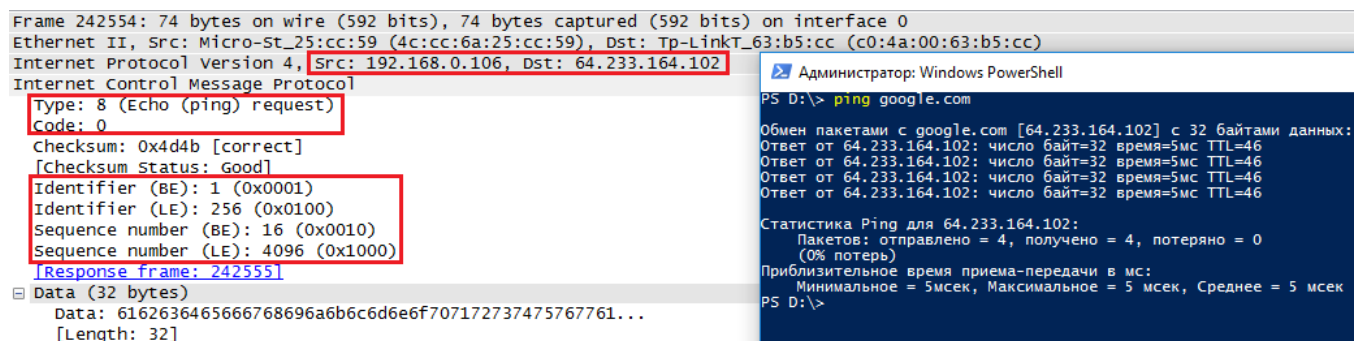


Рис. 1.2: ICMP эхо-запрос

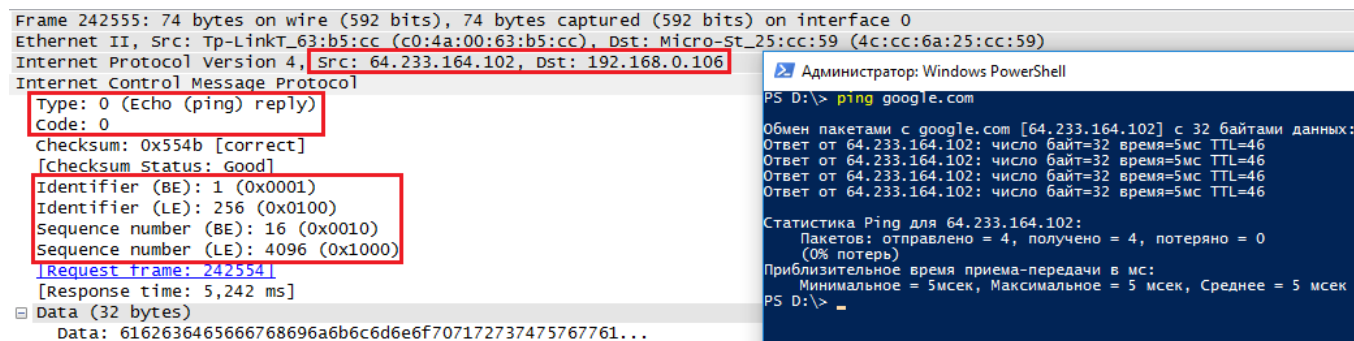


Рис. 1.3: ICMP эхо-ответ

Пакеты были распознаны как ICMP с пометкой "Echo (ping) reply/request" что означает эхо запрос/ответ. Графа Destination показывает IP адрес удаленного сервера, который мы пингуем, Source показывает IP адрес текущего компьютера.

Поля "Identifier" и "Sequence number" присутствуют только в эхо запросе/ответе (ICMP типы 0 и 8) и необходимы для сопоставления ответа и запроса.

## Ping с фрагментацией

Для фрагментации пакета необходимо явно указать его размер, превышающий MTU (maximum transmission unit) - максимальный размер полезного блока данных одного пакета, который может быть передан без фрагментации. Для интерфейса Ethernet II значение MTU равно 1500 байт. Тогда без учета заголовка (20 байт) длина одного фрагмента не превышает 1480 байт.

Трафик утилиты Ping с измененными параметрами (*bytes* = 4096, *TTL* = 61):

The image shows a Wireshark packet capture of a fragmented ICMP Echo (ping) request. The packet list shows three fragments: 911071 (1514 bytes), 911072 (1514 bytes), and 911073 (1178 bytes). The packet details for the third fragment (911073) are expanded, showing the IP header with 'Fragment offset: 2960' and the ICMP header with 'Type: 8 (Echo (ping) request)'. The packet bytes pane shows the raw data of the third fragment, including the IP header and the ICMP payload. To the right, a Windows PowerShell window shows the command 'ping google.com -l 4096' and its output, which indicates that the ping was successful with 4096 bytes of data and a TTL of 61.

```
Time Source Destination Protocol Length Info
911071 986.011606 192.168.0.106 80.70.231.42 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=549c) [Reassembled in
911072 986.011611 192.168.0.106 80.70.231.42 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=549c) [Reassembled in
911073 986.011613 192.168.0.106 80.70.231.42 ICMP 1178 Echo (ping) request id=0x0001, seq=17/4352, ttl=128 (reply in 911076)

Frame 911073: 1178 bytes on wire (9424 bits), 1178 bytes captured (9424 bits) on interface 0
Ethernet II, Src: Micro-St_25:cc:59 (4c:cc:6a:25:cc:59), Dst: Tp-LinkT_63:b5:cc (c0:4a:00:63:b5:cc)
Internet Protocol Version 4, Src: 192.168.0.106, Dst: 80.70.231.42
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1164
    Identification: 0x549c (21660)
  Flags: 0x00
  Fragment offset: 2960
  Time to live: 128
  Protocol: ICMP (1)
  Header checksum: 0x0000 [validation disabled]
  [Header checksum status: Unverified]
  Source: 192.168.0.106
  Destination: 80.70.231.42
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
  [3 IPv4 Fragments (4104 bytes): #911071(1480), #911072(1480), #911073(1144)]
    [Frame: 911071, payload: 0-1479 (1480 bytes)]
    [Frame: 911072, payload: 1480-2959 (1480 bytes)]
    [Frame: 911073, payload: 2960-4103 (1144 bytes)]
    [Fragment count: 3]
    [Reassembled IPv4 length: 4104]
    [Reassembled IPv4 data: 08009f94000100116162636465666768696a6b6c6d6e6f70...]
Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0x9f94 [correct]
  [Checksum status: Good]
  Identifier (BE): 1 (0x0001)
  Identifier (LE): 256 (0x0100)
  Sequence number (BE): 17 (0x0011)
  Sequence number (LE): 4352 (0x1100)
  [Response frame: 911076]
```

Рис. 1.4: Фрагментированный эхо-запрос (последний фрагмент)

Видно, что пакет разделился на три части. Стоит отметить, что фрагментация пакета осуществляется на уровне IP и каждый фрагмент имеет одинаковое значение поля "Identificaton".

The image shows a Wireshark packet capture of the first fragment of a fragmented ICMP Echo (ping) request. The packet list shows a single fragment: 911071 (1514 bytes). The packet details for the first fragment (911071) are expanded, showing the IP header with 'Total Length: 1500' and the ICMP header with 'Type: 8 (Echo (ping) request)'. The packet bytes pane shows the raw data of the first fragment, including the IP header and the ICMP payload. To the right, a Windows PowerShell window shows the command 'ping google.com -l 4096' and its output, which indicates that the ping was successful with 4096 bytes of data and a TTL of 61.

```
Time Source Destination Protocol Length Info
911071 986.011606 192.168.0.106 80.70.231.42 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=549c) [Reassembled in
911072 986.011611 192.168.0.106 80.70.231.42 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=549c) [Reassembled in
911073 986.011613 192.168.0.106 80.70.231.42 ICMP 1178 Echo (ping) request id=0x0001, seq=17/4352, ttl=128 (reply in 911076)

Frame 911071: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
Ethernet II, Src: Micro-St_25:cc:59 (4c:cc:6a:25:cc:59), Dst: Tp-LinkT_63:b5:cc (c0:4a:00:63:b5:cc)
Internet Protocol Version 4, Src: 192.168.0.106, Dst: 80.70.231.42
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1500
    Identification: 0x549c (21660)
  Flags: 0x01 (More Fragments)
  Fragment offset: 0
  Time to live: 128
  Protocol: ICMP (1)
  Header checksum: 0x0000 [validation disabled]
  [Header checksum status: Unverified]
  Source: 192.168.0.106
  Destination: 80.70.231.42
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
  Reassembled IPv4 in frame: 911073
Data (1480 bytes)
  Data: 08009f94000100116162636465666768696a6b6c6d6e6f70...
  [Length: 1480]
```

Рис. 1.5: Фрагментированный эхо-запрос (первый фрагмент)

Флаги, установленные в 0x1 свидетельствуют о наличии других фрагментов. Первый фрагмент определяется нулевым смещением.

## 1.4.2 Утилита Tracert

Tracert базируется на использовании поля TTL протокола ICMP. Первый пакет имеет TTL=1, для каждого последующего пакета TTL инкрементируется. Это продолжается до тех пор пока не придет эхо-ответ, а не ошибка истечения TTL. Трафик tracert это набор ICMP пакетов с типом "Time-to-live exceeded" (код 0x11) и в случае успеха, последний успешный эхо-ответ.

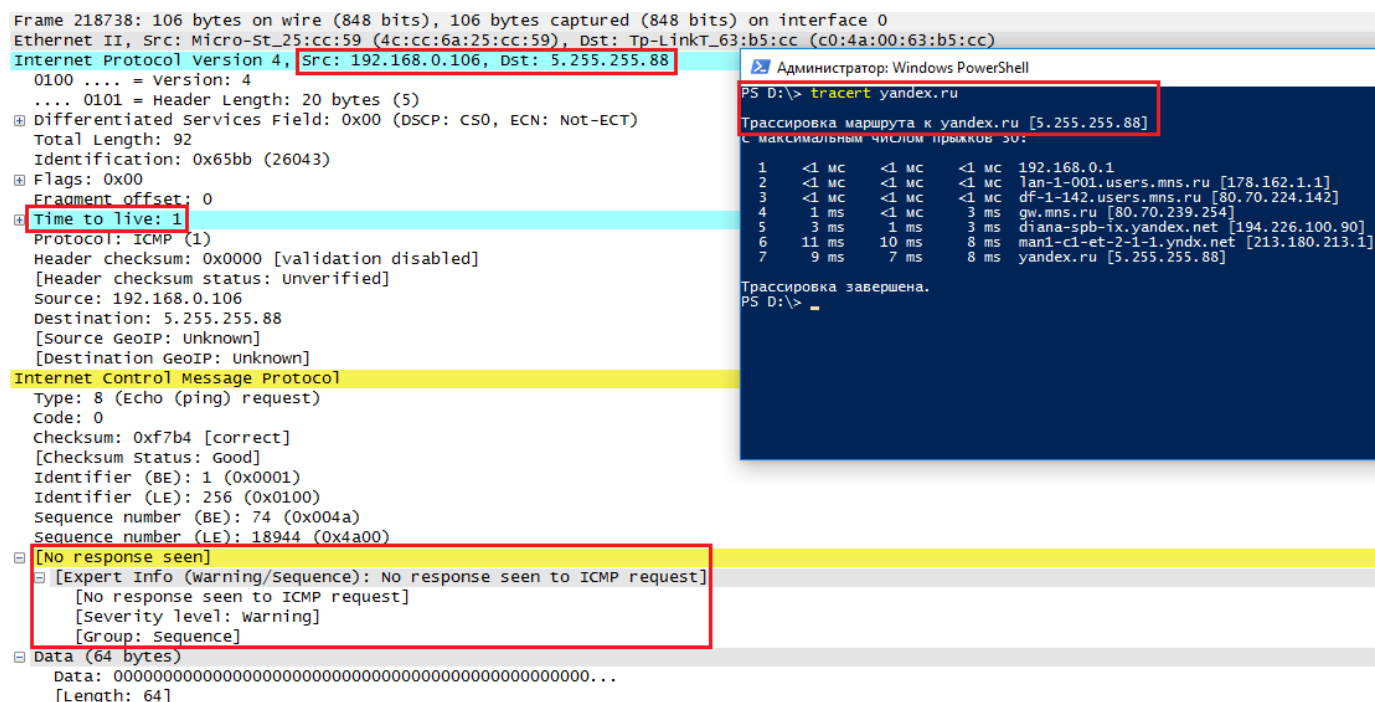


Рис. 1.6: Процесс трассировки yandex.ru (первый ICMP эхо-запрос)

Видно, что первый эхо-запрос имеет TTL равный единице, это означает, что на первом же маршрутизаторе, проверяющем значение TTL пакет будет уничтожен и вернется сообщение об ошибке.

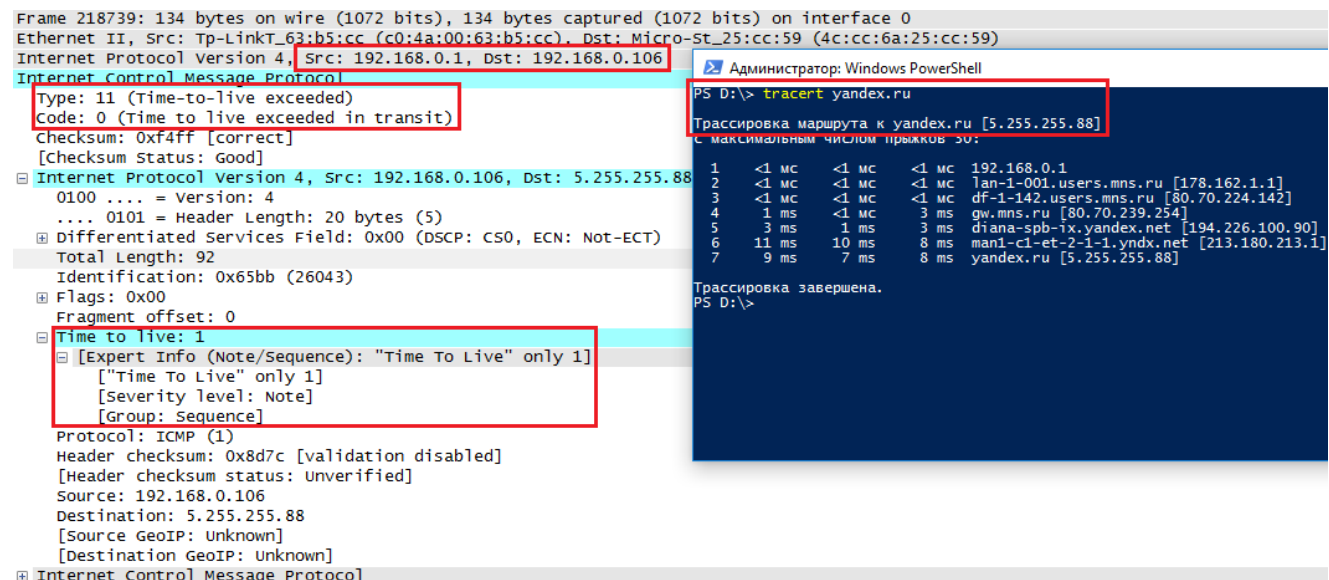


Рис. 1.7: Процесс трассировки yandex.ru (сообщение об ошибке Time-to-live exceeded)

Как и ожидалось, первая остановка это сетевой шлюз. В этом узле TTL стало равным нулю и был отправлен ICMP пакет с ошибкой типа "Time-to-live exceeded".

### 1.4.3 Протокол ICMP

ICMP ошибка о недоступности хоста (host unreachable) отправляется маршрутизатором, когда он получает IP датаграмму, которую невозможно перенаправить. Для получения данной ошибки в домашней сети будем пинговать тот локальный IP адрес, к которому шлюз по умолчанию не сможет проложить маршрут.

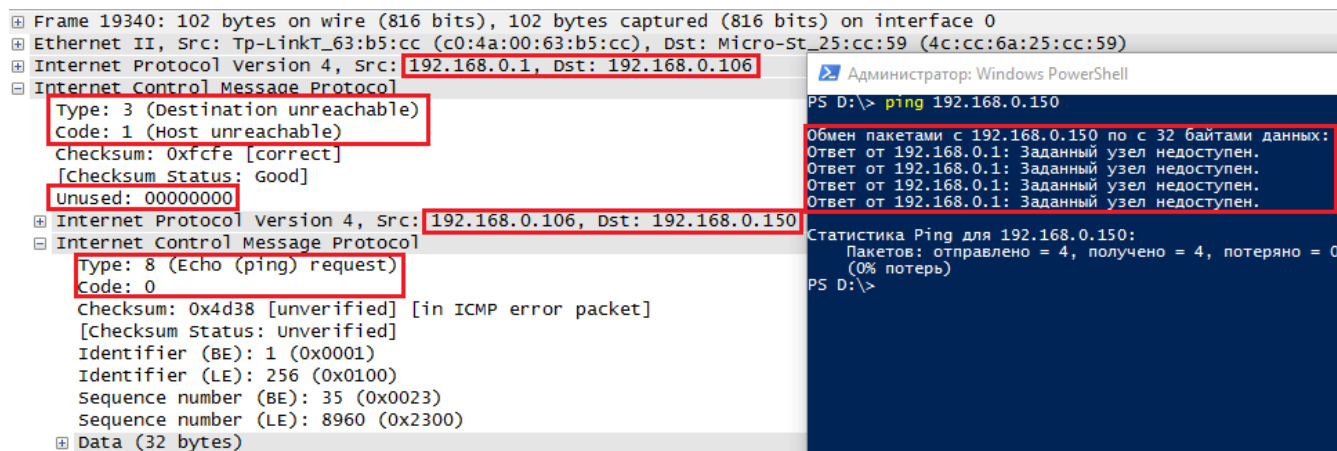


Рис. 1.8: ICMP ошибки недоступности адресата

В результате, на каждый посылаемый ICMP эхо запрос, вернулись ICMP пакеты с ошибкой "Destination unreachable (Host unreachable)" с типом 0x3 и кодом 0x1. Особенностью данной ошибки является то, что она посылается от шлюза, а не от узла назначения, что не удивительно, потому что к нему не удалось проложить маршрут. Также, при данном типе ошибки присутствует 32-битное не используемое поле, исходный IP заголовок, а также первые 64-бита исходной датаграммы.

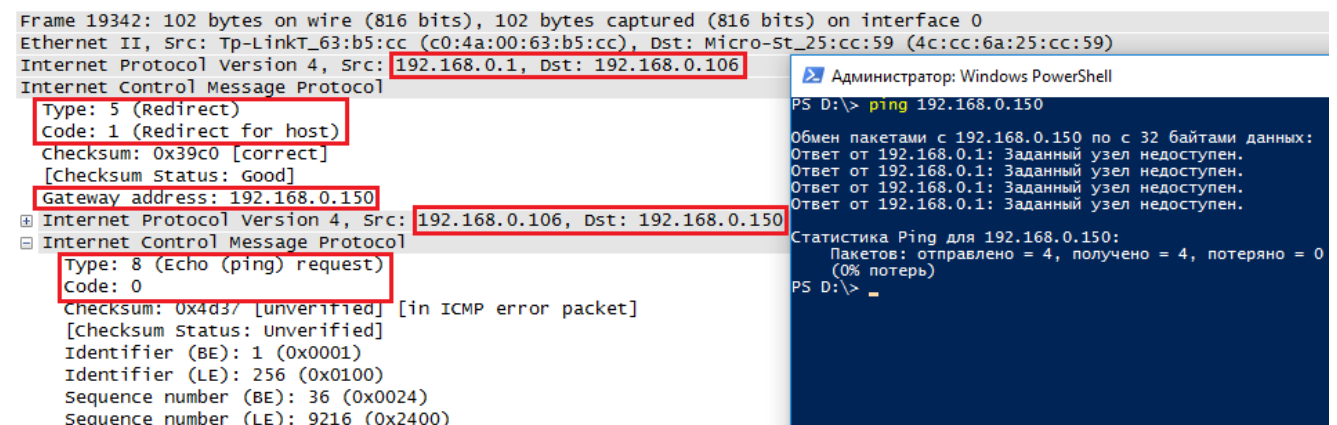


Рис. 1.9: ICMP информирование о перенаправлении

После ICMP пакета "Destination unreachable (Host unreachable)" следует ICMP пакет "Redirect (Redirect for host)" который информирует о том, что необходимо создать новый маршрут к указанному хосту и внести его в таблицу маршрутизации.



#### 1.4.4 Протокол ARP

Рассмотрим пару ARP пакетов, которая демонстрирует работу протокола.

```
Frame 448939: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
Ethernet II, Src: Micro-St_25:cc:59 (4c:cc:6a:25:cc:59), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: Micro-St_25:cc:59 (4c:cc:6a:25:cc:59)
  Sender IP address: 192.168.0.106
  Target MAC address: 00:00:00:00:00:00 (00:00:00:00:00:00)
  Target IP address: 192.168.0.1
```

Рис. 1.10: ARP запрос

Был отправлен широковещательный ARP запрос с заданным полем "Target IP Address" и нулевым полем "Target MAC Address".

```
Frame 448940: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet II, Src: Tp-LinkT_63:b5:cc (c0:4a:00:63:b5:cc), Dst: Micro-St_25:cc:59 (4c:cc:6a:25:cc:59)
Address Resolution Protocol (reply)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: reply (2)
  Sender MAC address: Tp-LinkT_63:b5:cc (c0:4a:00:63:b5:cc)
  Sender IP address: 192.168.0.1
  Target MAC address: Micro-St_25:cc:59 (4c:cc:6a:25:cc:59)
  Target IP address: 192.168.0.106
```

Рис. 1.11: ARP ответ

Был получен ARP ответ, в котором поле "Sender MAC Address" содержит искомый MAC адрес. Тип ARP пакета указывается в поле Opcode (запрос 0x1, ответ 0x2).

#### 1.4.5 Протокол TCP

##### Установление соединения

Попробуем установить TCP соединение с помощью утилиты telnet.

Для установления соединения посылается TCP пакет с управляющим битом SYN (синхронизация номеров последовательности) и номером последовательности. Сервер получает сегмент, запоминает номер последовательности и пытается создать сокет (буферы и управляющие структуры памяти) для обслуживания нового клиента. В случае успеха сервер посылает клиенту сегмент с номером последовательности и флагами SYN и ACK, и переходит в состояние SYN-RECEIVED. В случае неудачи сервер посылает клиенту сегмент с флагом RST.

В заголовке TCP пакета также присутствуют следующие поля:

- *Sequence Number* - Если установлен флаг SYN, то это изначальный порядковый номер — ISN (Initial Sequence Number), и первый байт данных, которые будут переданы в следующем пакете, будет иметь номер, равный  $ISN + 1$ . В противном случае, если SYN не установлен, первый байт данных, передаваемый в данном пакете, имеет этот порядковый номер.
- *Acknowledgment Number* - Если установлен бит 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.

В данном случае на сервер был отправлен пакет с флагом SYN и  $ISN=0$ . Был получен пакет с установленным флагом ACK и номером последовательности  $ACK=1$ , что означает, что пакет с  $ISN=0$  был успешно получен. Также в этом пакете установлен флаг SYN и  $ISN=0$ , что означает, что ожидается ACK со стороны клиента. Последний пакет содержит только  $ACK=1$  для сервера.

Ответ содержит установленные биты SYN и ACK, что свидетельствует об успешной установке соединения.



Time	Source	Destination	Protocol	Length	Info
14567	16.237300	192.168.0.106	46.255.138.1	TCP	66 54628→80 [SYN] Seq=0 win=8192 Len=0 MSS=1460
14886	16.373374	46.255.138.1	192.168.0.106	TCP	66 80→54628 [SYN, ACK] Seq=0 Ack=1 win=29200 Len=0
14887	16.373406	192.168.0.106	46.255.138.1	TCP	54 54628→80 [ACK] Seq=1 Ack=1 win=65536 Len=0

Frame 14567: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0  
 Ethernet II, Src: Micro-St\_25:cc:59 (4c:cc:6a:25:cc:59), Dst: Tp-LinkT\_63:b5:cc (c0:4a:00:63:b5:cc)  
 Internet Protocol Version 4, Src: 192.168.0.106, Dst: 46.255.138.1  
 Transmission Control Protocol, Src Port: 54628, Dst Port: 80, Seq: 0, Len: 0

Source Port: 54628  
 Destination Port: 80  
 [Stream index: 11]  
 [TCP Segment Len: 0]  
 Sequence number: 0 (relative sequence number)  
 Acknowledgment number: 0  
 Header Length: 32 bytes

Flags: 0x002 (SYN)

- 000. .... = Reserved: Not set
- ...0 .... = Nonce: Not set
- .... 0... = Congestion window Reduced (CWR): Not set
- .... .0.. = ECN-Echo: Not set
- .... ..0. = Urgent: Not set
- .... ...0 = Acknowledgment: Not set
- .... .... 0... = Push: Not set
- .... ..... 0... = Reset: Not set
- .... .... .1. = Syn: Set

[Expert Info (Chat/Sequence): Connection establish request (SYN): server port 80]  
 [Connection establish request (SYN): server port 80]  
 [Severity level: chat]  
 [Group: Sequence]

.... .... 0 = Fin: Not set  
 [TCP Flags: .....S.]  
 window size value: 8192  
 [Calculated window size: 8192]  
 Checksum: 0x7a39 [unverified]  
 [Checksum Status: unverified]  
 Urgent pointer: 0

Options: (12 bytes), Maximum segment size, No-operation (NOP), window scale, No-operation (NOP), No-operation (NOP), SACK permitted

Рис. 1.12: TCP запрос на установление соединения SYN

Time	Source	Destination	Protocol	Length	Info
14567	16.237300	192.168.0.106	46.255.138.1	TCP	66 54628→80 [SYN] Seq=0 win=8192 Len=0 MSS=1460
14886	16.373374	46.255.138.1	192.168.0.106	TCP	66 80→54628 [SYN, ACK] Seq=0 Ack=1 win=29200 Len=0
14887	16.373406	192.168.0.106	46.255.138.1	TCP	54 54628→80 [ACK] Seq=1 Ack=1 win=65536 Len=0

Frame 14886: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0  
 Ethernet II, Src: Tp-LinkT\_63:b5:cc (c0:4a:00:63:b5:cc), Dst: Micro-St\_25:cc:59 (4c:cc:6a:25:cc:59)  
 Internet Protocol Version 4, Src: 46.255.138.1, Dst: 192.168.0.106  
 Transmission Control Protocol, Src Port: 80, Dst Port: 54628, Seq: 0, Ack: 1, Len: 0

Source Port: 80  
 Destination Port: 54628  
 [Stream index: 11]  
 [TCP Segment Len: 0]  
 Sequence number: 0 (relative sequence number)  
 Acknowledgment number: 1 (relative ack number)  
 Header Length: 32 bytes

Flags: 0x012 (SYN, ACK)

- 000. .... = Reserved: Not set
- ...0 .... = Nonce: Not set
- .... 0... = Congestion window Reduced (CWR): Not set
- .... .0.. = ECN-Echo: Not set
- .... ..0. = Urgent: Not set
- .... ...0 = Acknowledgment: Set
- .... .... 0... = Push: Not set
- .... ..... 0... = Reset: Not set
- .... .... .1. = Syn: Set

[Expert Info (Chat/Sequence): Connection establish acknowledge (SYN+ACK): server port 80]  
 [Connection establish acknowledge (SYN+ACK): server port 80]  
 [Severity level: chat]  
 [Group: Sequence]

.... .... 0 = Fin: Not set  
 [TCP Flags: .....A..S.]  
 window size value: 29200  
 [Calculated window size: 29200]  
 Checksum: 0x320a [unverified]  
 [Checksum Status: unverified]  
 Urgent pointer: 0

Options: (12 bytes), Maximum segment size, No-operation (NOP), No-operation (NOP), SACK permitted, No-operation (NOP), window scale

Рис. 1.13: Удачная установка TCP соединения (ответ с сервера)

Time	Source	Destination	Protocol	Length	Info
14887	16.373406	192.168.0.106	46.255.138.1	TCP	54 54628→80 [ACK] Seq=1 Ack=1 win=65536 Len=0
151663	196.480174	46.255.138.1	192.168.0.106	TCP	60 80→54628 [FIN, ACK] Seq=1 Ack=1 win=29696 Len=0
151664	196.480195	192.168.0.106	46.255.138.1	TCP	54 54628→80 [ACK] Seq=1 Ack=2 win=65536 Len=0

Frame 151664: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0  
 Ethernet II, Src: Micro-St\_25:cc:59 (4c:cc:6a:25:cc:59), Dst: Tp-LinkT\_63:b5:cc (c0:4a:00:63:b5:cc)  
 Internet Protocol Version 4, Src: 192.168.0.106, Dst: 46.255.138.1  
 Transmission Control Protocol, Src Port: 54628, Dst Port: 80, Seq: 1, Ack: 2, Len: 0

Source Port: 54628  
 Destination Port: 80  
 [Stream index: 11]  
 [TCP Segment Len: 0]  
 Sequence number: 1 (relative sequence number)  
 Acknowledgment number: 2 (relative ack number)  
 Header Length: 20 bytes  
 Flags: 0x010 (ACK)  
 000. .... = Reserved: Not set  
 ...0 .... = Nonce: Not set  
 .... 0... = Congestion Window Reduced (cwr): Not set  
 .... 0.. = ECN-Echo: Not set  
 .... ..0. = Urgent: Not set  
 .... ...1 .... = Acknowledgment: Set  
 .... .... 0... = Push: Not set  
 .... .... ..0. = Reset: Not set  
 .... .... ..0. = Syn: Not set  
 .... .... ...0 = Fin: Not set  
 [TCP Flags: .....A....]  
 Window size value: 256  
 [Calculated window size: 65536]  
 [window size scaling factor: 256]  
 checksum: 0x7a2d [unverified]  
 [checksum status: Unverified]  
 Urgent pointer: 0

Рис. 1.14: Удачная установка TCP соединения (последний ACK)

Последний этап это отправка на сервер пакета с установленным флагом ACK, после чего соединение переходит в состояние ESTABLISHED.

### Неудачное соединение

Рассмотрим попытку неудачного соединения. Клиент посылает пакет с управляющим битом SYN (запрос на установление соединения рассмотрен в предыдущем пункте).

Time	Source	Destination	Protocol	Length	Info
5248	7.114887	192.168.0.106	192.168.0.104	TCP	66 54649→80 [SYN] Seq=0 win=8192 Len=0 MSS=1460 WS=256
5368	7.160942	192.168.0.104	192.168.0.106	TCP	60 80→54649 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
6174	7.776920	192.168.0.104	192.168.0.106	TCP	60 80→54649 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
6179	8.389807	192.168.0.104	192.168.0.106	TCP	60 80→54649 [RST, ACK] Seq=1 Ack=1 win=0 Len=0

Frame 5368: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0  
 Ethernet II, Src: OnepIusT\_46:46:fa (c0:ee:fb:46:46:fa), Dst: Micro-St\_25:cc:59 (4c:cc:6a:25:cc:59)  
 Internet Protocol Version 4, Src: 192.168.0.104, Dst: 192.168.0.106  
 Transmission Control Protocol, Src Port: 80, Dst Port: 54649, Seq: 1, Ack: 1, Len: 0

Source Port: 80  
 Destination Port: 54649  
 [Stream index: 7]  
 [TCP Segment Len: 0]  
 Sequence number: 1 (relative sequence number)  
 Acknowledgment number: 1 (relative ack number)  
 Header Length: 20 bytes  
 Flags: 0x014 (RST, ACK)  
 000. .... = Reserved: Not set  
 ...0 .... = Nonce: Not set  
 .... 0... = Congestion Window Reduced (cwr): Not set  
 .... 0.. = ECN-Echo: Not set  
 .... ..0. = Urgent: Not set  
 .... ...1 .... = Acknowledgment: Set  
 .... .... 0... = Push: Not set  
 .... .... ..1. = Reset: Set  
 [Expert Info (warning/Sequence): Connection reset (RST)]  
 [Connection reset (RST)]  
 [Severity level: warning]  
 [Group: Sequence]  
 .... .... ..0. = Syn: Not set  
 .... .... ...0 = Fin: Not set  
 [TCP Flags: .....A.R..]  
 Window size value: 0  
 [Calculated window size: 0]  
 [window size scaling factor: -1 (unknown)]  
 checksum: 0x4197 [unverified]  
 [checksum status: unverified]  
 Urgent pointer: 0

Telnet 192.168.0.104  
 Microsoft Telnet> o sakh.com 80  
 Подключение к sakh.com...  
 Подключение к узлу утеряно.  
 Microsoft Telnet> o 192.168.0.104 80  
 Подключение к 192.168.0.104...Не удалось открыть подключение.  
 Microsoft Telnet> o 192.168.0.104 80  
 Подключение к 192.168.0.104...Не удалось открыть подключение.  
 Microsoft Telnet>

Рис. 1.15: Неудачная попытка TCP соединения

Сервер посылает пакет с установленным управляющим битом RST. После чего клиент уже не пытается установить соединение.

## Завершение соединения

Рассмотрим процесс завершения соединения (инициированного с серверной стороны):

No.	Time	Source	Destination	Protocol	Length	Info
91032	118.860602	192.168.0.106	192.168.0.107	TCP	66	65100→33252 [FIN, ACK] Seq=83
91033	118.866137	192.168.0.107	192.168.0.106	TCP	66	33252→65100 [FIN, ACK] Seq=15
91034	118.866163	192.168.0.106	192.168.0.107	TCP	66	65100→33252 [ACK] Seq=84 Ack=

<
⊞ Frame 91032: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
⊞ Ethernet II, Src: Micro-St_25:cc:59 (4c:cc:6a:25:cc:59), Dst: LiteonTe_64:22:23 (74:de:2b:64:22:23)
⊞ Internet Protocol Version 4, Src: 192.168.0.106, Dst: 192.168.0.107
⊞ Transmission Control Protocol, Src Port: 65100, Dst Port: 33252, Seq: 83, Ack: 15, Len: 0
Source Port: 65100
Destination Port: 33252
[Stream index: 17]
[TCP Segment Len: 0]
Sequence number: 83 (relative sequence number)
Acknowledgment number: 15 (relative ack number)
Header Length: 32 bytes
⊞ Flags: 0x011 (FIN, ACK)
000. .... = Reserved: Not set
...0 .... = Nonce: Not set
.... 0... = Congestion window Reduced (CWR): Not set
.... 0... = ECN-Echo: Not set
.... ..0. = Urgent: Not set
.... ..1. = Acknowledgment: Set
.... .... 0... = Push: Not set
.... .... 0.. = Reset: Not set
.... .... ..0. = Syn: Not set
⊞ .... ..1 = Fin: Set
⊞ [Expert Info (chat/Sequence): Connection finish (FIN)]
[Connection finish (FIN)]
[Severity level: chat]
[Group: Sequence]
[TCP Flags: .....A...F]
window size value: 260
[Calculated window size: 66560]
[window size scaling factor: 256]
checksum: 0x824c [unverified]
[checksum status: Unverified]
urgent pointer: 0
⊞ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps

Рис. 1.16: Завершение TCP соединения

Если соединение уже было установлено, то завершение соединения производится следующим образом:

- Посылка серверу от клиента флага FIN на завершение соединения.
- Сервер посылает клиенту флаги ответа ACK, FIN, что соединение закрыто.
- После получения этих флагов клиент закрывает соединение и в подтверждение отправляет серверу ACK, что соединение закрыто.

В данном случае был запущен сервер по адресу 192.168.0.107:65100, после чего к нему подключился клиент. Клиент был принудительно отключен со стороны сервера: клиенту был отправлен пакет с установленными битами FIN и ACK, после чего клиент отправляет такой же пакет на сервер и переходит из состояния ESTABLISHED в состояние CLOSE WAIT, в завершении сервер отправляет клиенту ACK и переходит в состояние CLOSED, клиент получает ACK и также переходит в состояние CLOSED.

## Установка соединения с закрытым портом

При попытке подключения к отсутствующему порту, не приходит ACK и RST, поэтому клиент находится в подвешенном состоянии и ожидает ответа. Такое поведение обусловлено работой межсетевого экрана, который не позволяет узнать извне какие порты открыты. Результат подключения к закрытому порту с отключенным межсетевым экраном рассмотрен в приложении 1.

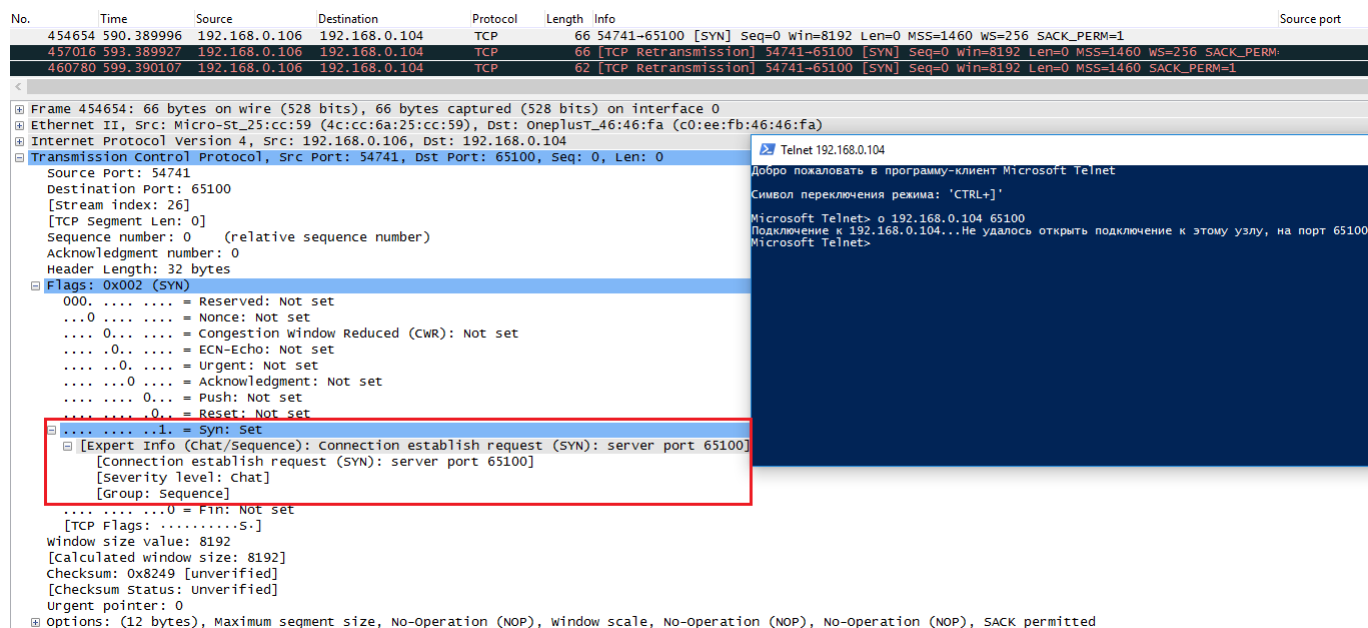


Рис. 1.17: Ожидание ответа на SYN

Посылается несколько запросов на установление соединения с некоторым таймаутом, после чего клиент считает что хост недоступен.

## 1.5 Вывод

В ходе работы был исследован сетевой трафик утилит ping и tracert а также протоколов ICMP, ARP и TCP.

На практике не имеет особого смысла исследовать трафик утилит ping и tracert, потому что они представляют полную и наглядную информацию о сетевом взаимодействии непосредственно внутри консоли. Однако, для выяснения причины ошибки соединения или для определения адресов и портов назначения исследование сетевого трафика подходит отлично.

Также необходимость анализа сетевого трафика обусловлена тем, что клиент-серверные приложения чаще всего не предоставляют полную информацию об используемом трафике. Анализаторы трафика по типу Wireshark имеют широкие возможности для фильтрации трафика, что позволяет просматривать сетевой трафик конкретных приложений.

## 1.6 Приложение 1

### 1.6.1 Установка соединения с закрытым портом при отключенном межсетевом экране

Попробуем подключиться к закрытому порту компьютера с ОС Ubuntu по адресу 192.168.0.108:300 с компьютера с адресом 192.168.0.106. По умолчанию ОС Ubuntu имеет стандартный межсетевой экран ufw, который не позволяет узнать извне, какие порты открыты или закрыты (рис. 1.17). Для того, чтобы разрешить взаимодействие с портом 300 извне, воспользуемся утилитой iptables:

```
sudo iptables -I INPUT -p tcp --dport 300 -m state --state NEW -j ACCEPT
```

Результат подключения к закрытому порту:

No.	Time	Source	Destination	Protocol	Length	Info
26465	34.759172	192.168.0.106	192.168.0.108	TCP	66	59952 → 300 [SYN] Seq=1825502112 win=8192
26466	34.760931	192.168.0.108	192.168.0.106	TCP	60	300 → 59952 [RST, ACK] Seq=0 Ack=1825502113
⊕ Frame 26466: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0						
⊕ Ethernet II, Src: LiteonTe_64:22:23 (74:de:2b:64:22:23), Dst: Micro-St_25:cc:59 (4c:cc:6a:25:cc:59)						
⊕ Internet Protocol Version 4, Src: 192.168.0.108, Dst: 192.168.0.106						
⊖ Transmission Control Protocol, Src Port: 300, Dst Port: 59952, Seq: 0, Ack: 1825502113, Len: 0						
Source Port: 300						
Destination Port: 59952						
[Stream index: 7]						
[TCP Segment Len: 0]						
Sequence number: 0						
Acknowledgment number: 1825502113						
Header Length: 20 bytes						
⊕ Flags: 0x014 (RST, ACK)						

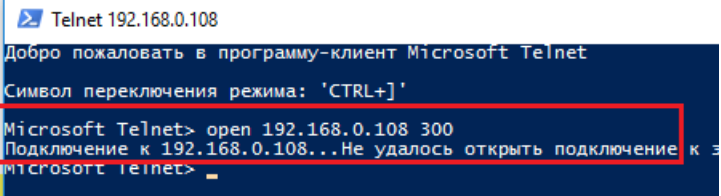


Рис. 1.18: Результат подключения к закрытому порту

При попытке подключения к закрытому порту ожидаемо вернулся пакет с установленным флагом RST.