Национальный исследовательский университет информационных технологий, механики и оптики
Кафедра вычислительной техники
Сети ЭВМ и телекоммуникации

Лабораторная работа №3
«Аналииз структуры сетевого трафика с помощью программ Wireshark и Ostinato»
Вариант 5

Студентка:
Куклина М., P3301
Преподаватель:
Шинкарук Д.Н.

Санкт-Петербург, 2017

Цели работы

1. Исследование структуры сетевых пакетов с помощью анализатора трафика Wireshark.
2. Исследование структуры сетевых пакетов с помощью генератора пакетов Ostinato.

Часть 1. Исследование структуры сетевых пакетов с помощью анализатора трафика Wireshark

Протокол IP

Конечный узел: mk.ru.

Анализ трафика производился на операционной системе семейства Linux, на которой аналогом требуемой в задании утилиты `tracert` служит команда `traceroute -icmp`.

Структура первого пакета.

```
IP header
----------
Version:         4
IHL:             5
DSCP:            0
ECN:             0
Total length:    60
Identification:  0x2c80
Flags:           0x00
Fragment offset: 0
TTL:             1
Protocol:        1
Header Checksum: 0x49eb
Source IP:       192.168.1.26
Destination IP:  92.242.36.162
```

Из заголовка следует, что:

- исходный адрес хоста: 192.168.1.26 – адрес компьютера в локальной сети, с которого отправился пакет;
- протокол верхнего уровня определяется полем IP-заголовка «Protocols», которое имеет значение 1, обозначающее протокол ICMP в соответствии с RFC-790 ;
- размер заголовка IP определяется полем «Internet Header Length», значение в котором равно 5 DWORD'ам или 20 байтам;
- размер пакета, включающий заголовок и данные, определяется полем «Total Length», следовательно, данные занимают 40 байт;
- поле «TTL» равняется 1, что означает максимальное количество узлов на пути следования пакета; в данном случае ровно на первом узле пакет уничтожится, и в ответ от текущего узле придёт ICMP Time Exceeded с "информацией об узле";
- поле «Identification» идентифицирует отдельный пакет; используется при фрагемнтации: фрагменты с одинаковыми ID собираются в один пакет, порядок фрагментов определяется полем «Fragment offset», а наличие фрагметов – полем «Flags».

Фрагментация пакетов

Конечный узел: wireshark.com

При запуске команды анализатор трафика выдаёт следующие результаты.

```
IP header 1                          IP header 2
-----------                          -----------
Version:         4                   Version:      4
IHL:             5                   IHL:          5
DSCP:            0                   DSCP:         0
ECN:             0                   ECN:          0
Total length:    1500                Total length: 548
Identification:  0x0d6b              ID:           0x0d6b
Flags:           0x01                Flags:        0x00
Fragment offset: 0                   Frag. off:    1480
TTL:             64                  TTL:          64
Protocol:        1                   Protocol:     1
Header Checksum: 0x43c5              Checksum:     0x66c4
Source IP:       192.168.1.26        Source IP:    192.168.1.26
Destination IP:  172.110.10.86       Dest. IP:     172.110.10.86
```

Из этих двух последовательно пойманных пакетов видно, что:

- имеет место фрагментация пакетов;
- первый пакет является фрагементом, чему свидетельствуют значение поля «Flags» (0x01 – More fragments), второй пакет является конечным (0x00 в поле флагов и ненулевое значение в поле «Fragment offset»);
- из всего указанного следует, что пакета всего два.

Протокол ICMP

Конечный узел: wireshark.com

Часть 1. Исследование с помощью команды ping

После запуска команды анализатор трафика выдал следующие результаты.

```
IP header                            IP header
---------                            ---------
Version:        4                    Version:      4
IHL:            5                    IHL:          5
Total Length:   84                   Total Length: 84
ID:             0x5afb               ID:           0x5afb
Frag.off.:      0                    Frag.off.:    0
TTL:            64                   TTL:          40
Protocol:       1                    Protocol:     1
Checksum:       0x6727               Checksum:     0x66a1
Source IP:      192.168.1.26         Source IP:    172.110.10.86
Destination IP: 172.110.10.86        Dest. IP:     192.168.1.26

ICMP Echo                            ICMP Echo Reply
---------                            ---------------
Type:      8                         Type:     0
Code:      0                         Code:     0
Checksum:  0x3848                    Checksum: 0x4048
ID:        0x09b5                    ID:       0x09b5
Seq. num.: 1                         Seq. num: 1
```

1. Программа захватила 20 пакетов: ping отправил 10 ICMP Echo Request и на каждый получил ICMP Echo Reply.
2. IP адрес источника: 192.168.1.26; IP адресанта: 104.25.218.21.

3. Анализ первого пакета. Тип пакета ICMP определяется полями «Type» и «Code»; в данном случае пара (8,0) определяет Echo Request. Поля «ID» и «Sequence number» одинаковы в двух пакетах и служат для опередеделения соответствия пары запрос-ответ. Также ID не меняется при всей ping-сессии для её идентификации. Значение seq инкрементируется с каждым отправленым ICMP Echo Request. Поля «Type» и «Code» занимают 1 байт каждое; «Cheksum», «Identifier» и «Sequence number» – 2 байта.
4. Анализ второго пакета. Всё отличие от первого пакета обнаруживается в поле «Type», которое в паре с полем «Code» определяют Echo Reply.

Часть 2. Исследование с помощью команды traceroute

```
IP header
---------
Version:        4
IHL:            5
Total length:   60
ID:             0x3995
Flags:          0x00
Frag. off:      0
TTL:            1
Protocol:       1
Checksum:       0x07a6
Source:         192.168.1.26
Dest:           172.110.10.86

ICMP Echo
---------
Type:    8
Code:    0
Cheksum: 0x07a6
ID:      0x1536
Seq.num: 1
```

Данный пакет отличается от пакета ICMP Echo.

- Полем Total Length:
- Полем TTL:

```
IP header
---------
Version:        4
IHL:            5
Total length:   88
ID:             0xa1f7
Flags:          0x00
Frag. off:      0
TTL:            64
Protocol:       1
Checksum:       0x07a6
Source:         192.168.1.1
Dest:           192.168.1.26

ICMP Time Exceeded
------------------
Type:    11
Code:    0
Cheksum: 0x0f4ff
+ Unused
+ Old IP header and 64 bits of datagram.
```

В ответ на ICMP Echo из-за истечения TTL в ответ с узла шлётся ICMP Time Exceeded, определяемый значением 11 в поле «Type». Помимо полей «Code» и «Checksum» пакет содержит в себе IP заголовок и первые 64 бита датаграммы пакета, TTL которого достиг нуля на данном узле.

Первый полученный ICMP Echo Reply.

```
IP header
---------
Version:        4
IHL:            5
Total length:   60
ID:             0xaa64
Flags:          0x00
Frag. off:      0
TTL:            49
Protocol:       1
Checksum:       0x66b6
Source:         172.110.10.86
Dest:           192.168.1.26

ICMP Echo Reply
---------------
Type:    0
Code:    0
Cheksum: 0x7519
ID:      5430
Seq.num: 43
```

Пакет ICMP Echo Reply отличается от ICMP Time Exceeded как отличаются друг от друга грозовые и перистые облака: ICMP Time Exceeded отправляется с узлов, на которых обнуляется TTL, если пакет достиг адреса назначения, то с хоста отправляется ICMP Echo Reply. В данном случае с хоста отправилось пять пакетов в ответ на запрос с значениями seq от 40 до 44 и TTL – 14 и 15. Как только ответ дошёл до источника, запросы прекратились.

Часть 2. Исследование структуры сетевых пакетов с помощью генератора пакетов Ostinato

Часть A

ARP

ARP (Address Resolution Protocol) – протокол сетевого уровня, использующийся для определения аппаратного адреса по данному протокольному в Ethernet сетях.

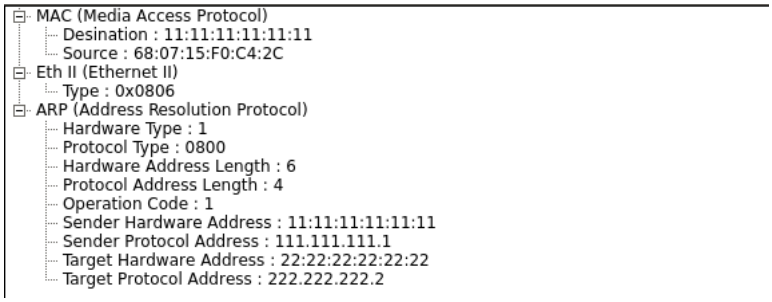

Рис. 1. Конфигурация пакета ARP в Ostinato

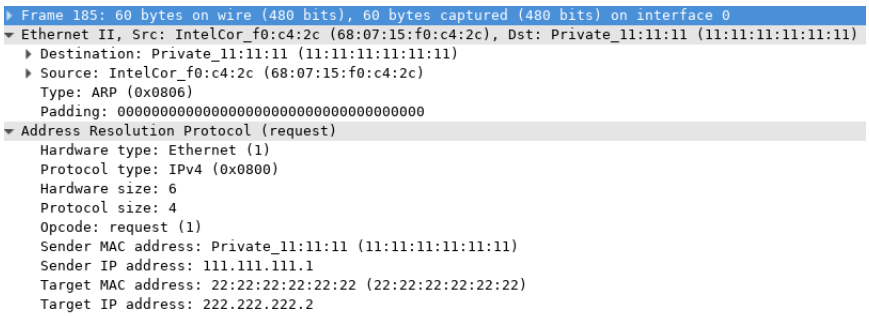

Рис. 2. ARP в Wireshark

UDP

UDP (User Datagram Protocol) – протокол транспортного уровня, использующийся для обмена датаграммами без установления соединения.

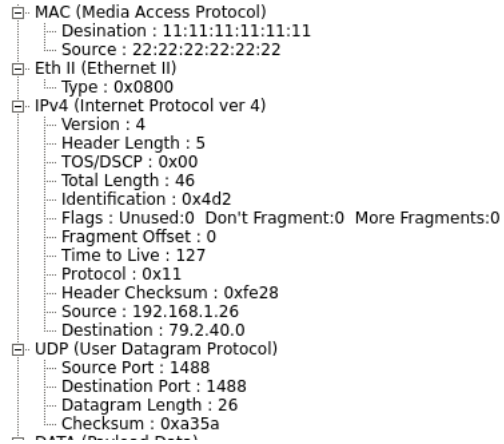

Рис. 3. Конфигурация пакета UDP в Ostinato

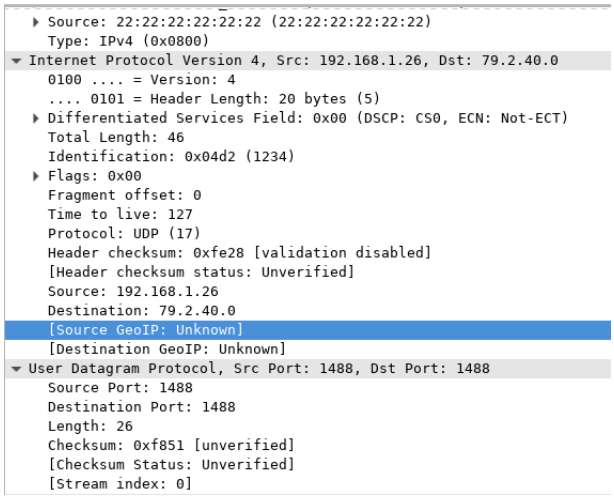

Рис. 4. UDP в Wireshark

Конфигурации трафика

Рассмотрим трафик с двумя режимами.

Пойманный Wireshark трафик говорит нам о том, что Ostinato действует в соответствии с документацией.

177	21.98183143	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18
181	22.18183100	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18
182	22.38183293	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18
183	22.58189064	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18
184	22.78187566	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18
185	22.78188540	IntelCor_f0:c4:2c	Private_11:11:11	ARP	60 Who has 222.222.222.2? Tell 111.111.111.1
186	22.98189514	IntelCor_f0:c4:2c	Private_11:11:11	ARP	60 Who has 222.222.222.2? Tell 111.111.111.1
190	23.18191560	IntelCor_f0:c4:2c	Private_11:11:11	ARP	60 Who has 222.222.222.2? Tell 111.111.111.1
191	23.38191149	IntelCor_f0:c4:2c	Private_11:11:11	ARP	60 Who has 222.222.222.2? Tell 111.111.111.1

Рис. 5. Sequential Streams

35	4.396319639	IntelCor_f0:c4:2c	Private_11:11:11	ARP	60 Who has 222.222.222.2? Tell 111.111.111.1
36	4.596313786	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18
37	4.596320459	IntelCor_f0:c4:2c	Private_11:11:11	ARP	60 Who has 222.222.222.2? Tell 111.111.111.1
38	4.796314291	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18
39	4.796321575	IntelCor_f0:c4:2c	Private_11:11:11	ARP	60 Who has 222.222.222.2? Tell 111.111.111.1
40	4.996316383	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18
41	4.996324175	IntelCor_f0:c4:2c	Private_11:11:11	ARP	60 Who has 222.222.222.2? Tell 111.111.111.1
45	5.196316058	192.168.1.26	79.2.40.0	UDP	60 1488 → 1488 Len=18

Рис. 6. Interleaved Streams

Часть Б

IPv4

Генерация пакетов IPv4.

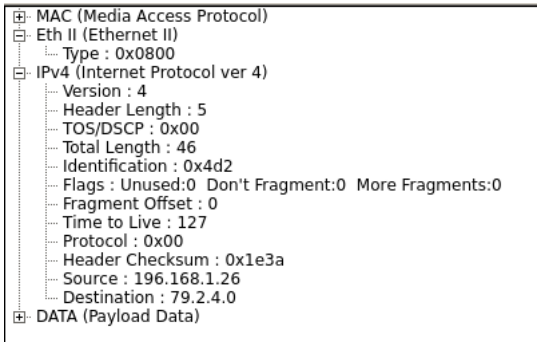

Рис. 7. Конфигурация пакета IP в Ostinato

```
▸ Frame 12: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▾ Ethernet II, Src: 22:22:22:22:22:22 (22:22:22:22:22:22), Dst: Private_11:11:11 (11:11:11:11:11:11)
  ▸ Destination: Private_11:11:11 (11:11:11:11:11:11)
  ▸ Source: 22:22:22:22:22:22 (22:22:22:22:22:22)
    Type: IPv4 (0x0800)
▾ Internet Protocol Version 4, Src: 196.168.1.26, Dst: 79.2.4.0
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 46
    Identification: 0x04d2 (1234)
  ▸ Flags: 0x00
    Fragment offset: 0
    Time to live: 127
    Protocol: IPv6 Hop-by-Hop Option (0)
    Header checksum: 0x1e3a [validation disabled]
    [Header checksum status: Unverified]
    Source: 196.168.1.26
    Destination: 79.2.4.0
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
▸ Data (26 bytes)
```

Рис. 8. IP в Wireshark

ICMP

Генерация пакетов ICMPv4.

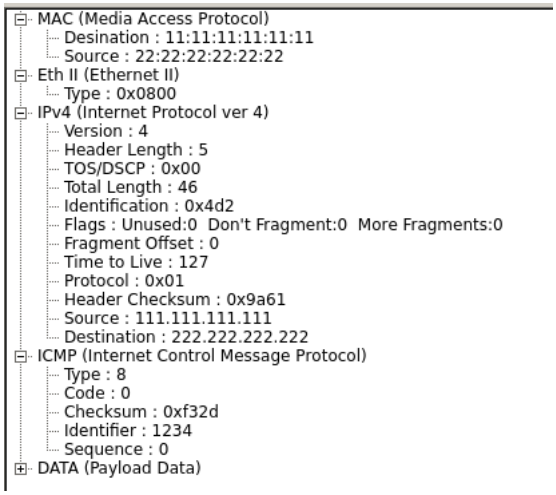

Рис. 9. Конфигурация пакета ICMP в Ostinato

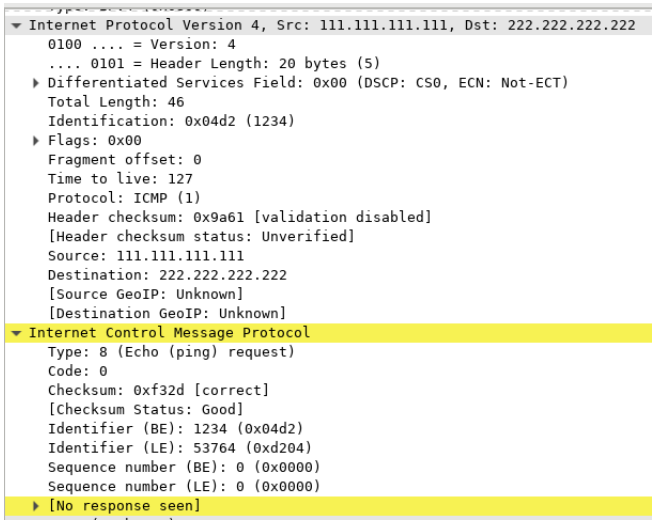

Рис. 10. ICMP в Wireshark

Конфигурации трафика

Рассмотрим трафик с двумя режимами.

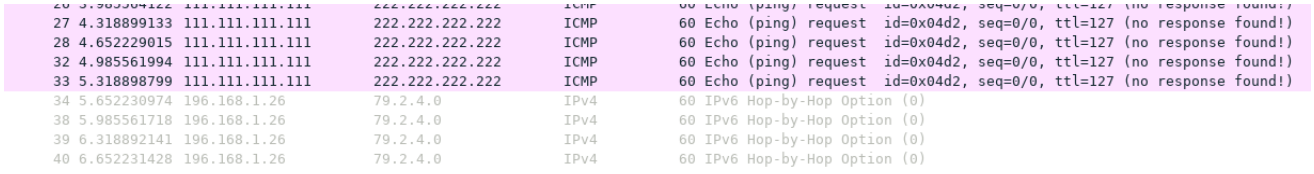

Рис. 11. Sequential Streams

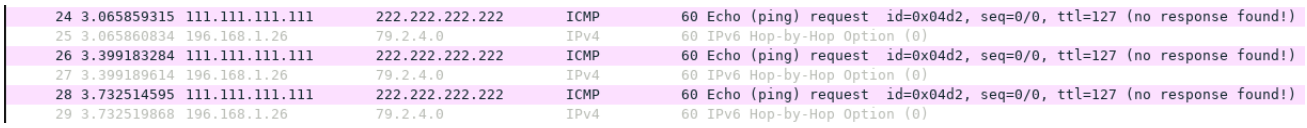

Рис. 12. Interleaved Streams

Пойманный Wireshark трафик говорит нам о том, что Ostinato действует в соответствии с документацией.

Вывод

При выполнении лабораторной работы производилось исследование структуры сетевых пакетов с помощью Wireshark и Ostinato. В результате выполнения работы были сделаны следующие выводы и подмечены следующие детали.

1. `traceroute` на каждый TTL отправляет три пакета в силу не совсем очевидных причин; скорее всего, для точности результатов и повышения вероятности ответа.
2. Есть несколько методов работы traceroute: помимо ICMP-метода существует UDP-подход, при котором при запросе отсылается простой UDP-пакет с TTL=1 и максимальным возможным значением порта (32767); разработчики утверждают, что они надеются[1], что узел не примет target-порт и в ответ отошлёт ICMP Destination Unreachable. Данный метод испольуется при предположении, что ICMP Echo Server может быть не доступен на узле (к примеру, отключён администратором[1]).
3. Для ответа на ICMP Echo каждый узел согласно RFC-1122 должен реализовать ICMP Echo Server.

Список литературы

1. «Comparing ICMP and UDP Traceroute Methods», ftp://ftp.hp.com/pub/hpcp/UDP-ICMP-Traceroutes.pdf
2. https://tools.ietf.org/html/