МИНИСТЕРСТВО НАУКИ
И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

НГТУ НЭТИ | Факультет прикладной математики и информатики

Кафедра прикладной математики

Лабораторная работа № 4

по дисциплине «Операционные системы и компьютерные сети»

АНАЛИЗ СТРУКТУРЫ ФРЕЙМА ТЕХНОЛОГИИ ETHERNET

Бригада 9

Группа ПМ-24 ГЕРАСИМЕНКО ВАДИМ

ПАРАСКУН ИВАН

Преподаватели СИВАК МАРИЯ АЛЕКСЕЕВНА

КОБЫЛЯНСКИЙ ВАЛЕРИЙ ГРИГОРЬЕВИЧ

Новосибирск, 2024

1. Цель работы

Спроектировать и реализовать программу, выполняющую анализ структуры кадра/фрейма технологии Ethernet.

2. Постановка задачи

Разработать и отладить программу, выполняющую анализ потока кадров. Потоки кадров представлены в виде файлов двоичного формата, в кадрах отсутствует преамбула и контрольная сумма, для исходящего кадра длина может быть меньше минимальной. Необходимо выполнить обработку файла с именем ethers09.bin. При выполнении работы в дистанционном режиме в обязательном порядке выполнить анализ файлов ethers06.bin и ethers07.bin.

Требования к программе:

• предусмотреть возможность ввода имени файла с клавиатуры;

• обеспечить вывод на экран по каждому кадру: номер, размер, тип, IP- адресов (основную информацию заголовка IP-пакета), MAC- адресов (основную информацию заголовка кадра); также необходимо вывести итоговые результаты обработки: общее число обработанных кадров и число кадров каждого типа.

3. Текст программы

Main class

```
public class Main {
    public static void main(String[] args) throws FileNotFoundException {
        Scanner scanner = new Scanner(System.in);

        final String currentFolderPath = "src/ru/nstu/networks/ether-
net_frame_analysis";
        final String binaryExtension = ".bin";

        File folder = new File(currentFolderPath + "/resources");
        List<String> fileNames = Arrays.stream(Objects.re-
quireNonNull(folder.listFiles()))
                .filter(file -> file.getName().endsWith(binaryExtension))
                .sorted(Comparator.comparing(File::getName))
                .map(File::getName).toList();

        if (fileNames.isEmpty()) {
            throw new FileNotFoundException("The resource folder does not
contain files for analysis");
        }

        System.out.println("Files available for analysis:");
        fileNames.forEach(System.out::println);
```

```
        File file;

        while (true) {
            try {
                System.out.printf("%nEnter file name: ");
                String fileName = scanner.nextLine();

                if (!fileNames.contains(fileName)) {
                    throw new FileNotFoundException("The resources folder
does not contain a file called \""
                            + fileName + "\"");
                }

                file = new File(currentFolderPath + "/resources/" + file-
Name);
                break;
            } catch (FileNotFoundException e) {
                System.out.println("Error: " + e.getMessage() + ". Try
again");
            }
        }

        System.out.println("Where to output the result?");
        System.out.println("Enter '1' - to print the results to the con-
sole");
        System.out.println("Enter any other character - to output the re-
sults to the 'out.txt' file.");

        PrintStream printStream = Objects.equals(scanner.nextLine(), "1")
                ? System.out
                : new PrintStream(new FileOutputStream(currentFolderPath
+ "/out.txt"));
        System.out.printf("Let's launch...%n%n");
        EthernetFrameAnalyzer.analyze(file, printStream);
        System.out.println("Ready!");
    }
}
```

EthernetFrameAnalyzer class

Статические константы, определенные для данного класса:

```
public static final String BINARY_EXTENSION = ".bin";

public static final String ETHERNET_2_DIX = "Ethernet II (DIX)";
public static final String LLC = "Ethernet 802.2 (LLC)";
public static final String SNAP = "Ethernet 802.2 (SNAP)";
public static final String NOVELL = "Ethernet 802.3 (Raw/Novell)";
public static final int SNAP_CODE = 0x00AA;
public static final int NOVELL_CODE = 0xFFFF;
public static final int ETHERNET_802_2_CODE_SIZE = 0x0002;
public static final String ICMP = "ICMP";
public static final String TCP = "TCP";
public static final String UDP = "UDP";
public static final int ICMP_NUMBER = 0x0001;
public static final int TCP_NUMBER = 0x0006;
public static final int UDP_NUMBER = 0x0011;

public static final String IP_VERSION_4 = "IPv4";
public static final String ARP = "ARP";
public static final String RARP = "RARP";
public static final int IP_VERSION_4_NUMBER = 0x0800;
public static final int ARP_NUMBER = 0x0806;
public static final int RARP_NUMBER = 0x0835;
public static final String REQUEST = "Request";
public static final String REPLY = "Reply";
public static final int REQUEST_NUMBER = 0x0001;
public static final int REPLY_NUMBER = 0x0002;

public static final int MAX_FRAME_SIZE = 0xFFFF;
public static final int MAC_ADDRESS_SIZE = 0x0006;
public static final int TYPE_OR_LENGTH_SIZE = 0x0002;
public static final int VERSION_AND_HEADER_LENGTH_SIZE = 0x0001;

public static final int VERSION_BITS_COUNT = 0x0004;
public static final int WORD_32_BITS_SIZE_IN_BYTES = 0x0004;

public static final int TOS_SIZE = 0x0001;
public static final int TTL_SIZE = 0x0001;
public static final int IP_SIZE = 0x0004;

public static final String ETHERNET_HARDWARE_TYPE = "Ethernet";
public static final int ETHERNET_HARDWARE_TYPE_NUMBER = 0x0001;
public static final int HARDWARE_ADDRESS_LENGTH_SIZE = 0x0001;
public static final int HARDWARE_TYPE_SIZE = 0x0002;
public static final int PROTOCOL_ADDRESS_LENGTH_SIZE = 0x0001;
public static final int PROTOCOL_TYPE_SIZE = 0x0002;

public static final int DATAGRAM_LENGTH_SIZE = 0x0002;
public static final int DATAGRAM_FIELD_MAX_SIZE = 0x05DC;

public static final int IDENTIFIER_SIZE = 0x0002;
public static final int OPTIONS_AND_FRAGMENT_OFFSET_SIZE = 0x0002;
public static final int UPPER_LEVEL_PROTOCOL_SIZE = 0x0001;
public static final int HEADER_CHECKSUM_SIZE = 0x0002;
public static final int OPERATION_SIZE = 0x0002;
```

Ключевой статический метод класса:

```
public static void analyze(File binaryFile, PrintStream out) {
    validateBinaryFile(binaryFile);

    try (BufferedInputStream stream = new BufferedInputStream(new
FileInputStream(binaryFile))) {
        int fileSize = (int) binaryFile.length();
        out.println("File name: " + binaryFile.getName());
        out.printf("File size: %d%n", fileSize);

        int framesCount = 0;
        int dixFramesCount = 0;
        int novellFramesCount = 0;
        int snapFramesCount = 0;
        int llcFramesCount = 0;
        int ip4FramesCount = 0;
        int arpFramesCount = 0;
        int rarpFramesCount = 0;

        for (int readBytesCount = 0; readBytesCount < fileSize; )
{
            int startIndex = readBytesCount;
            int index = startIndex;

            ++framesCount;
            out.printf("%nFRAME #%d%n", framesCount);

            byte[] bytes = new byte[MAX_FRAME_SIZE];

            readBytesCount += read(stream, bytes, index, MAC_AD-
DRESS_SIZE);
            out.println("MAC (receiver): " + getMacAddress(bytes,
index));
            index += MAC_ADDRESS_SIZE;

            readBytesCount += read(stream, bytes, index, MAC_AD-
DRESS_SIZE);
            out.println("MAC (sender): " + getMacAddress(bytes,
index));
            index += MAC_ADDRESS_SIZE;

            readBytesCount += read(stream, bytes, index,
TYPE_OR_LENGTH_SIZE);
            int dataFieldSize = convert2BytesToInt(bytes, index);
            index += TYPE_OR_LENGTH_SIZE;

            if (dataFieldSize > DATAGRAM_FIELD_MAX_SIZE) {
                out.println("Frame type: " + ETHERNET_2_DIX);
                ++dixFramesCount;

                switch (dataFieldSize) {
                    case IP_VERSION_4_NUMBER:
```

```
                        int headerStartIndex = index;
                        readBytesCount += read(stream, bytes, in-
dex, VERSION_AND_HEADER_LENGTH_SIZE);
                        String versionAndHeaderLength = con-
vertByteToBin(bytes[index]);
                        index += VERSION_AND_HEADER_LENGTH_SIZE;

                        out.println("Protocol type: " + IP_VER-
SION_4);
                        ++ip4FramesCount;

                        int headerLength = Integer.parseInt(ver-
sionAndHeaderLength.substring(VERSION_BITS_COUNT), 2)
                                * WORD_32_BITS_SIZE_IN_BYTES;
                        out.println("Header length: " + header-
Length + " bytes");

                        readBytesCount += read(stream, bytes, in-
dex, TOS_SIZE);
                        out.println("Type of service: " + con-
vertByteToInt(bytes[index]));
                        index += TOS_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, DATAGRAM_LENGTH_SIZE);
                        int dataLength = convert2BytesToInt(bytes,
index);
                        out.println("Data length: " + dataLength +
" bytes");
                        index += DATAGRAM_LENGTH_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, IDENTIFIER_SIZE + OPTIONS_AND_FRAGMENT_OFFSET_SIZE);
                        //identifier, options and fragment offset
skipped
                        index += IDENTIFIER_SIZE + OP-
TIONS_AND_FRAGMENT_OFFSET_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, TTL_SIZE);
                        out.println("TTL: " + con-
vertByteToInt(bytes[index]));
                        index += TTL_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, UPPER_LEVEL_PROTOCOL_SIZE);
                        out.println("Protocol: " + getProto-
colName(bytes, index));
                        index += UPPER_LEVEL_PROTOCOL_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, HEADER_CHECKSUM_SIZE);
```

```
                            out.println("Header checksum: " + con-
vert2BytesToInt(bytes, index));
                            index += HEADER_CHECKSUM_SIZE;

                            readBytesCount += read(stream, bytes, in-
dex, IP_SIZE);
                            out.println("IP (sender): " + getIpAd-
dress(bytes, index));
                            index += IP_SIZE;

                            readBytesCount += read(stream, bytes, in-
dex, IP_SIZE);
                            out.println("IP (receiver): " + getIpAd-
dress(bytes, index));
                            index += IP_SIZE;

                            int unreadHeaderSize = index - startIndex
                                    - headerLength - 2 * MAC_AD-
DRESS_SIZE - TYPE_OR_LENGTH_SIZE;

                            if (unreadHeaderSize != headerStartIndex)
{
                                //options skipped
                                readBytesCount += read(stream, bytes,
index, unreadHeaderSize);
                                index += unreadHeaderSize;
                            }

                            readBytesCount += read(stream, bytes, in-
dex, dataLength - headerLength);
                            out.println("Data: " + concate-
nateData(bytes, index, readBytesCount));
                            break;
                        case ARP_NUMBER, RARP_NUMBER:
                            if (dataFieldSize == ARP_NUMBER) {
                                out.println("Protocol type: " + ARP);
                                ++arpFramesCount;
                            } else {
                                out.println("Protocol type: " + RARP);
                                ++rarpFramesCount;
                            }

                            readBytesCount += read(stream, bytes, in-
dex, HARDWARE_TYPE_SIZE);
                            int hardwareTypeNumber = con-
vert2BytesToInt(bytes, index);
                            index += HARDWARE_TYPE_SIZE;

                            String hardwareType = hardwareTypeNumber
== ETHERNET_HARDWARE_TYPE_NUMBER
                                    ? ETHERNET_HARDWARE_TYPE
                                    : hardwareTypeNumber + " (missing
from the database)";
```

```
                        out.println("Hardware type: " + hardware-
Type);

                        readBytesCount += read(stream, bytes, in-
dex, PROTOCOL_TYPE_SIZE);
                        int protocolTypeNumber = con-
vert2BytesToInt(bytes, index);
                        String protocolType = protocolTypeNumber
== IP_VERSION_4_NUMBER
                                ? IP_VERSION_4
                                : protocolTypeNumber + " (missing
from the database)";
                        out.println("Protocol type: " + proto-
colType);
                        index += PROTOCOL_TYPE_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, HARDWARE_ADDRESS_LENGTH_SIZE);
                        int hardwareAddressLength = con-
vertByteToInt(bytes[index]);
                        out.println("Hardware address length: " +
hardwareAddressLength + " bytes");
                        index += HARDWARE_ADDRESS_LENGTH_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, PROTOCOL_ADDRESS_LENGTH_SIZE);
                        int protocolAddressLength = con-
vertByteToInt(bytes[index]);
                        out.println("Protocol address length: " +
protocolAddressLength + " bytes");
                        index += PROTOCOL_ADDRESS_LENGTH_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, OPERATION_SIZE);
                        int operationNumber = con-
vert2BytesToInt(bytes, index);
                        String operation = switch (operation-
Number) {
                            case REQUEST_NUMBER -> REQUEST;
                            case REPLY_NUMBER -> REPLY;
                            default -> operationNumber + " (miss-
ing from the database)";
                        };
                        out.println("Operation type: " + opera-
tion);
                        index += OPERATION_SIZE;

                        readBytesCount += read(stream, bytes, in-
dex, hardwareAddressLength);
                        out.println("Hardware address (sender): "
+ getMacAddress(bytes, index));
                        index += hardwareAddressLength;
```

```
                        readBytesCount += read(stream, bytes, in-
dex, protocolAddressLength);
                        out.println("Protocol address (sender): "
+ getIpAddress(bytes, index));
                        index += protocolAddressLength;

                        readBytesCount += read(stream, bytes, in-
dex, hardwareAddressLength);
                        out.println("Hardware address (receiver):
" + getMacAddress(bytes, index));
                        index += hardwareAddressLength;

                        readBytesCount += read(stream, bytes, in-
dex, protocolAddressLength);
                        out.println("Protocol address (receiver):
" + getIpAddress(bytes, index));
                        break;
                    default:
                        out.println("Protocol type number " + da-
taFieldSize + " (missing from the database)");
                    }
            } else {
                readBytesCount += read(stream, bytes, index,
ETHERNET_802_2_CODE_SIZE);
                int dataFirst2Bytes = convert2BytesToInt(bytes,
index);

                switch (dataFirst2Bytes) {
                    case NOVELL_CODE:
                        ++novellFramesCount;
                        out.println("Protocol type: " + NOVELL);
                        break;
                    case SNAP_CODE:
                        ++snapFramesCount;
                        out.println("Protocol type: " + SNAP);
                        break;
                    default:
                        ++llcFramesCount;
                        out.println("Protocol type: " + LLC);
                }

                out.println("Data length: " + dataFieldSize + "
bytes");

                readBytesCount += read(stream, bytes, index, data-
FieldSize - ETHERNET_802_2_CODE_SIZE);
                out.println("Data: " + concatenateData(bytes, in-
dex - ETHERNET_802_2_CODE_SIZE,
                        index + dataFieldSize - ETHER-
NET_802_2_CODE_SIZE));
            }
        }
```

```
            out.printf("%n----------------------------------------");
            out.printf("%nFrames count: %d%n", framesCount);
            out.printf("%n[Types]%n");
            out.println(LLC + ": " + llcFramesCount);
            out.println(SNAP + ": " + snapFramesCount);
            out.println(NOVELL + ": " + novellFramesCount);
            out.println(ETHERNET_2_DIX + ": " + dixFramesCount);
            out.println("\t" + IP_VERSION_4 + ": " + ip4FramesCount);
            out.println("\t" + ARP + ": " + arpFramesCount);
            out.println("\t" + RARP + ": " + rarpFramesCount);
            out.println("----------------------------------------");
        } catch (IOException e) {
            out.println("Error reading file \"" + binaryFile.getName()
+ "\": " + e.getMessage());
        }
}
```

Вспомогательные методы класса EthernetFrameAnalyzer:

```
private static int read(InputStream stream, byte[] array, int startIndex,
int length) throws IOException {
    int readBytesCount;
    int offset = startIndex;
    int end = length + startIndex;

    while (offset != end) {
        readBytesCount = stream.read(array, offset, length);
        offset += readBytesCount;
    }

    return offset - startIndex;
}

private static String convertByteToHex(byte number) {
    char[] hexDigits = new char[2];
    hexDigits[0] = Character.forDigit((number >> 4) & 0xF, 16);
    hexDigits[1] = Character.forDigit((number & 0xF), 16);
    return new String(hexDigits).toUpperCase();
}

private static String convertByteToBin(byte number) {
    return String.format("%8s", Integer.toBinaryString(number &
0xFF)).replace(' ', '0');
}

private static int convertByteToInt(byte number) {
    return number & 0xFF;
}

private static int convert2BytesToInt(byte[] array, int startIndex) {
    final int intLength = 4;
    final int dataLength = 2;
    byte[] bytes = new byte[intLength];

    System.arraycopy(array, startIndex, bytes, intLength - dataLength,
dataLength);
    return ByteBuffer.wrap(bytes).getInt();
}

private static String concatenateData(byte[] array, int startIndex, int
endIndex) {
    StringBuilder sb = new StringBuilder();
    int index = startIndex;

    while (index < endIndex) {
        sb.append(convertByteToHex(array[index]));
        ++index;
    }

    return sb.toString();
}

private static String getIpAddress(byte[] array, int startIndex) {
    StringBuilder sb = new StringBuilder();
    final char separator = '.';
```

```
    for (int i = 0; i < IP_SIZE; ++i) {
        int number = array[startIndex + i];
        sb.append(number & 0xFF).append(separator);
    }

    return sb.deleteCharAt(sb.length() - 1).toString();
}

private static String getMacAddress(byte[] array, int startIndex) {
    StringBuilder sb = new StringBuilder();
    final char separator = '-';

    for (int i = 0; i < MAC_ADDRESS_SIZE; ++i) {
        sb.append(convertByteToHex(array[i + startIndex])).append(separa-
tor);
    }

    return sb.deleteCharAt(sb.length() - 1).toString();
}

private static String getProtocolName(byte[] bytes, int startIndex) {
    return switch (bytes[startIndex]) {
        case ICMP_NUMBER -> ICMP;
        case TCP_NUMBER -> TCP;
        case UDP_NUMBER -> UDP;
        default -> bytes[startIndex] + " (missing from the database)";
    };
}

private static void validateBinaryFile(File binaryFile) {
    if (binaryFile == null) {
        throw new NullPointerException("File references to null");
    }

    String fileName = binaryFile.getName();

    if (!fileName.endsWith(BINARY_EXTENSION)) {
        throw new IllegalArgumentException("File must have a binary ex-
tension. "
                + "Current extension: " +
fileName.substring(fileName.lastIndexOf(".")));
    }
}
```

4. Анализ потока кадров

Файл *ethers09.bin:*

```
File name: ethers09.bin
File size: 1904

FRAME #1
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 52 bytes
TTL: 64
Protocol: TCP
Header checksum: 41706
IP (sender): 195.62.2.11
IP (receiver): 81.181.78.206
Data:
00168C2CE897EC01D3B5EBA08010BF2831F200000101080A3522BCDF0896050B

FRAME #2
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 139 bytes
TTL: 64
Protocol: TCP
Header checksum: 30964
IP (sender): 195.62.2.11
IP (receiver): 118.165.75.86
Data:
0019EC61E865E0E62899F2CF5018C0D82AD00000323230207368616D616E2E696E
746572666163652E6E736B2E73752045534D54502053656E646D61696C20382E31
342E332F382E31342E333B204D6F6E2C203230204F6374203230303820313A34
333A3538202B3037303020284E4F565354290D0A

FRAME #3
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 104 bytes
TTL: 51
Protocol: TCP
Header checksum: 62303
```

```
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
8C2C0016D3B5EBA0E897EC0180180045B5E200000101080A089605493522BCDFD0
1F729E16AF36EE5320F5232AAE45382A38A873F3641D0A606E70116EB5351E29CE
9C01D49095AC5FCE884B2C3F56EED34C2BE3

FRAME #4
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 104 bytes
TTL: 64
Protocol: TCP
Header checksum: 41653
IP (sender): 195.62.2.11
IP (receiver): 81.181.78.206
Data:
00168C2CE897EC01D3B5EBD48018BF28F30800000101080A3522BCED0896054940
069D58DACF56C4FEC91B353661F70A74143EC709C047605239DD741B25BB4D16AD
BCA75CEC1CD212C33CEF9BF73F4D3D912816

FRAME #5
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 116 bytes
TTL: 118
Protocol: TCP
Header checksum: 63224
IP (sender): 84.108.20.237
IP (receiver): 195.62.2.11
Data:
0B5500192A86DDF6E8681C995018FECA52C700004D41494C2046524F4D3A3C7879
676E666471786940626F6F6C617A2E636F6D3E0D0A5243505420544F3A203C6761
6C696E6140696E746572666163652E6E736B2E73753E0D0A444154410D0A

FRAME #6
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 40 bytes
TTL: 64
Protocol: TCP
```

```
Header checksum: 8952
IP (sender): 195.62.2.11
IP (receiver): 84.108.20.237
Data: 00190B55E8681C992A86DE425010C494A3640000

FRAME #7
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 67 bytes
TTL: 255
Protocol: UDP
Header checksum: 24357
IP (sender): 195.62.2.11
IP (receiver): 192.31.80.30
Data:
ADEB0035002FE6363FDC0000000100000000000106626F6F6C617A03636F6D0000
1C0001000029100000000000000

FRAME #8
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 136 bytes
TTL: 51
Protocol: TCP
Header checksum: 62270
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
8C2C0016D3B5EBD4E897EC3580180045B14400000101080A0896056E3522BCED32
FCF5E135361CADE8CEEB602ADD8F22E1876E75C8CE7BBF061E0DEE3DA9BE622F10
1963BFE57C373D34EB4F390B02078587F8F7D2FC089677B75CBCF855600E8FE6A6
52F8B00A203FE1274DB795B36633B9E2C7

FRAME #9
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 136 bytes
TTL: 64
Protocol: TCP
Header checksum: 41620
IP (sender): 195.62.2.11
```

```
IP (receiver): 81.181.78.206
Data:
00168C2CE897EC35D3B5EC288018BED4C00F00000101080A3522BCFE0896056EF2
C1A3C9EF6BEA01508068966D8B62098D2B5124B2BC27A6965CA12F9D46D7334273
A05CF7B3726F718988A9BD64A5FEDB23F1644BA86C276B9AFADBE366C7DFBE54BE
308F2CCE162A14F5635107D3187CDC3861

FRAME #10
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 104 bytes
TTL: 51
Protocol: TCP
Header checksum: 62301
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
8C2C0016D3B5EC28E897EC89801800454F3600000101080A089605973522BCFEB8
E48FA571567A8C3A2DE77F87DD039AF1C38023B300AD1B361C7DE89B80808B65B9
1EFE9D6F11A438208F44674A96BB9CB68615

FRAME #11
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 52 bytes
TTL: 51
Protocol: TCP
Header checksum: 62352
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
8C2C0016D3B5EC5CE897EC8980110045EEE500000101080A089605973522BCFE

FRAME #12
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 52 bytes
TTL: 64
Protocol: TCP
Header checksum: 41703
IP (sender): 195.62.2.11
```

```
IP (receiver): 81.181.78.206
Data:
00168C2CE897EC89D3B5EC5D8010BED4304700000101080A3522BD0D08960597

FRAME #13
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 60 bytes
TTL: 51
Protocol: TCP
Header checksum: 65041
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
8D010016D38AE64A00000000A00216D076490000020405B40402080A0896059700
00000001030307

FRAME #14
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 64 bytes
TTL: 64
Protocol: TCP
Header checksum: 41690
IP (sender): 195.62.2.11
IP (receiver): 81.181.78.206
Data:
00168D01E89DD0F2D38AE64BB012C0500EF800000101080A3522BD0D0896059702
0405B40103030001010402

FRAME #15
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 52 bytes
TTL: 64
Protocol: TCP
Header checksum: 41701
IP (sender): 195.62.2.11
IP (receiver): 81.181.78.206
Data:
00168C2CE897EC89D3B5EC5D8011BED4304600000101080A3522BD0D08960597
```

```
FRAME #16
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 151 bytes
TTL: 55
Protocol: UDP
Header checksum: 28111
IP (sender): 192.31.80.30
IP (receiver): 195.62.2.11
Data:
0035ADEB0083B4CC3FDC800000100000002000306626F6F6C617A03636F6D0000
1C0001C00C000200010002A3000015046E7334330D646F6D61696E636F6E74726F
6CC013C00C000200010002A3000007046E733434C02DC028000100010002A30000
04D845B916C049000100010002A3000004D06DFF160000290200000000000000

FRAME #17
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 71 bytes
TTL: 61
Protocol: UDP
Header checksum: 38186
IP (sender): 195.62.1.65
IP (receiver): 195.62.2.11
Data:
0035CA000033EF66E5AF818200010000000000002343403313535023138023539
07696E2D6164647204617270610000C0001

FRAME #18
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 67 bytes
TTL: 255
Protocol: UDP
Header checksum: 36651
IP (sender): 195.62.2.11
IP (receiver): 216.69.185.22
Data:
ADEB0035002F65E13F13000000010000000000106626F6F6C617A03636F6D0000
1C000100002910000000000000
```

```
FRAME #19
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 71 bytes
TTL: 255
Protocol: UDP
Header checksum: 35507
IP (sender): 195.62.2.11
IP (receiver): 217.71.128.65
Data:
CA0300350033DADCE5AF0100000100000000000023434033135350231380235390
07696E2D6164647204617270610000C0001

-------------------------------------
Frames count: 19

[Types]
Ethernet 802.2 (LLC): 0
Ethernet 802.2 (SNAP): 0
Ethernet 802.3 (Raw/Novell): 0
Ethernet II (DIX): 19
    IPv4: 19
    ARP: 0
    RARP: 0
-------------------------------------
```

Файл *ethers06.bin:*

```
File name: ethers06.bin
File size: 1118

FRAME #1
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 204 bytes
TTL: 51
Protocol: TCP
Header checksum: 38933
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
842A0016D2CDBDEEE837DA558018003AC53700000101080A0895F5313522B66E00
00009406142B7A99AC622BA22BAB3348B66531D16B0000001A6469666669652D68
656C6C6D616E2D67726F7570312D736861310000000773736 82D7273610000000A
6165733132382D6362630000000A6165733132382D6362630000009686D61632D
736861310000009686D61632D736861310000000046E6F6E65000000046E6F6E65
00000000000000000000000000000000000000000000

FRAME #2
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 52 bytes
TTL: 64
Protocol: TCP
Header checksum: 41829
IP (sender): 195.62.2.11
IP (receiver): 81.181.78.206
Data:
0016842AE837DA55D2CDBE868010BFB88FAD00000101080A3522B67E0895F531

FRAME #3
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-D5
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C000000000800000044D8AB0C08023000014
0002000F00

FRAME #4
MAC (receiver): 00-90-27-A1-36-D0
```

```
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 196 bytes
TTL: 51
Protocol: TCP
Header checksum: 38940
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
842A0016D2CDBE86E837DA558018003AB30D00000101080A0895F5573522B67E00
00008C051E0000008100D65D55739AC8A1AF24ABA8F0D2C3D28AF4758F37A878FD
E6F3A1732BB79A3DB3F4CE1BA4CC648BA54836DD1947ECB5B29C27FB77B40826B6
02CD3EA2BD2EA7DB24DE1EAA2DD7B1C77B06956587F6BDC8F135672C605771CD35
8954A979365A6614FF4420CF4404659D2B7769D7AB-
BAB0EA1B4B5E29F805E15C2309F4A7FEE7260000000000

FRAME #5
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-C3
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C1000000008000000044D8AB0C1800F000014
0002000F00

FRAME #6
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-C4
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C1000000008000000044D8AB0C18010000014
0002000F00

FRAME #7
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-C5
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C1000000008000000044D8AB0C18011000014
0002000F00

FRAME #8
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-C8
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C1000000008000000044D8AB0C18014000014
```

```
0002000F00

FRAME #9
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-C9
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C10000000800000044D8AB0C18016000014
0002000F00

FRAME #10
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-CA
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C10000000800000044D8AB0C18017000014
0002000F00

FRAME #11
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-CB
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C10000000800000044D8AB0C18018000014
0002000F00

FRAME #12
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-CC
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C10000000800000044D8AB0C18019000014
0002000F00

FRAME #13
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-CE
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C10000000800000044D8AB0C1801B000014
0002000F00

FRAME #14
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-CF
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
```

```
00260300000000000800000044D8AB0C10000000800000044D8AB0C1801C000014
0002000F00

FRAME #15
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-D0
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
00260300000000000800000044D8AB0C10000000800000044D8AB0C1801D000014
0002000F00

-------------------------------------
Frames count: 15

[Types]
Ethernet 802.2 (LLC): 12
Ethernet 802.2 (SNAP): 0
Ethernet 802.3 (Raw/Novell): 0
Ethernet II (DIX): 3
    IPv4: 3
    ARP: 0
    RARP: 0
-------------------------------------
```

Файл *ethers07.bin:*

```
File name: ethers07.bin
File size: 2763

FRAME #1
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 139 bytes
TTL: 64
Protocol: TCP
Header checksum: 55614
IP (sender): 195.62.2.11
IP (receiver): 62.167.64.216
Data:
00196EB7E83396CF12E6B4CE5018C0D887F500003232302073686l6D616E2E696E
746572666163652E6E736B2E73752045534D54502053656E646D61696C20382E31
342E332F382E31342E333B204D6F6E2C203230204F6374203230303820313A34
333A3439202B3037303020284E4F565354290D0A

FRAME #2
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 52 bytes
TTL: 51
Protocol: TCP
Header checksum: 29732
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
88330016D340918DE86CF7428010002E500F00000101080A0895FCBF3522B974

FRAME #3
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 72 bytes
TTL: 51
Protocol: TCP
Header checksum: 29711
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
```

```
Data:
88330016D340918DE86CF7428018002EA7660000101080A0895FCBF3522B97453
53482D322E302D6C69627373682D302E310D0A

FRAME #4
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 52 bytes
TTL: 64
Protocol: TCP
Header checksum: 41771
IP (sender): 195.62.2.11
IP (receiver): 81.181.78.206
Data:
00168833E86CF742D34091A18010C0508FC900000101080A3522B9830895FCBF

FRAME #5
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 796 bytes
TTL: 64
Protocol: TCP
Header checksum: 41026
IP (sender): 195.62.2.11
IP (receiver): 81.181.78.206
Data:
00168833E86CF742D34091A18018C050A18A00000101080A3522B9850895FCBF00
0002E40A14AF2873A32EC1C72343FA854535F5B9DD0000007E6469666669652D68
656C6C6D616E2D67726F75702D65786368616E67652D736861323536 2C64696666
69652D68656C6C6D616E2D67726F75702D65786368616E67652D736861312C6469
666669652D68656C6C6D616E2D67726F757031342D736861312C6469666669652D
68656C6C6D616E2D67726F7570312D736861310000000F7373682D7273612C7373
682D6473730000009D6165733132382D6362632C336465732D6362632C626C6F77
666973682D6362632C6361737431323 82D6362632C617263666F75723132382C61
7263666F75723235362C617263666F75722C6165733139322D6362632C61657332
35362D6362632C72696A6E6461656C2D636263406C797361746F722E6C69752E73
652C6165733132382D6374722C6165733139322D6374722C6165733235362D6374
720000009D6165733132382D6362632C336465732D6362632C626C6F7766697368
2D6362632C6361737431323 82D6362632C617263666F75723132382C617263666F
75723235362C617263666F75722C6165733139322D6362632C6165733235362D63
62632C72696A6E6461656C2D636263406C797361746F722E6C69752E73652C6165
733132382D6374722C6165733139322D6374722C6165733235362D637472000000
55686D61632D6D64352C686D61632D736861312C686D61632D726970656D643136
302C686D61632D726970656D64313630406F70656E7373682E636F6D2C686D6163
2D736861312D39362C686D61632D6D64352D393600000055686D61632D6D64352C
```

```
686D61632D736861312C686D61632D726970656D643136302C686D61632D726970
656D64313630406F70656E7373682E636F6D2C686D61632D736861312D39362C68
6D61632D6D64352D3936000000156E6F6E652C7A6C6962406F70656E7373682E63
6F6D000000156E6F6E652C7A6C6962406F70656E7373682E636F6D000000000000
0000000000000000000000000000000000000000

FRAME #6
MAC (receiver): 00-08-02-8F-DA-6E
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 238 bytes
TTL: 105
Protocol: TCP
Header checksum: 26384
IP (sender): 205.188.9.82
IP (receiver): 195.62.2.42
Data:
144608A77E66321E2F17E58D50184000154D00002A02198B00C00003000B00009A
5CAA650931303930393932383738000000A0001000200500000C0025C0A81E1F0000
0469040008E979C8E70000005000000037FFFFFFF00030A0C5AFEC0DE0000000A
0004D41107060006000400010000000500043 2C9A9000019000C1349134E134C13
43134B1344000D0030563FC8090B6F41BD9F79422609DFA2F31A093C6CD7FD4EC5
9D51A6474E34F5A04D6972616E64614D0007060000030A0C001D00090001000502
01D20472000F0004000054CA0003000448FBE1F3

FRAME #7
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 204 bytes
TTL: 51
Protocol: TCP
Header checksum: 29578
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
88330016D34091A1E86CFA2A8018003ADC5600000101080A0895FCE93522B98500
0000940614EDFA313163CCB6D5A84DA770A49CABB40000001A6469666669652D68
656C6C6D616E2D67726F7570312D736861310000000773736 82D7273610000000A
6165733132382D6362630000000A6165733132382D6362630000000 9686D61632D
736861310000000 9686D61632D736861310000000 46E6F6E65000000046E6F6E65
0000000000000000000000000000000000000000

FRAME #8
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
```

```
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 52 bytes
TTL: 64
Protocol: TCP
Header checksum: 41769
IP (sender): 195.62.2.11
IP (receiver): 81.181.78.206
Data:
00168833E86CFA2AD34092398010BFB88CA600000101080A3522B9940895FCE9

FRAME #9
MAC (receiver): FF-FF-FF-FF-FF-FF
MAC (sender): 00-08-02-8F-DA-6E
Frame type: Ethernet II (DIX)
Protocol type: ARP
Hardware type: Ethernet
Protocol type: IPv4
Hardware address length: 6 bytes
Protocol address length: 4 bytes
Operation type: Request
Hardware address (sender): 00-08-02-8F-DA-6E
Protocol address (sender): 195.62.2.42
Hardware address (receiver): 00-00-00-00-00-00
Protocol address (receiver): 195.62.2.1

FRAME #10
MAC (receiver): 00-08-02-8F-DA-6E
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: ARP
Hardware type: Ethernet
Protocol type: IPv4
Hardware address length: 6 bytes
Protocol address length: 4 bytes
Operation type: Reply
Hardware address (sender): 00-02-16-09-FA-40
Protocol address (sender): 195.62.2.1
Hardware address (receiver): 00-08-02-8F-DA-6E
Protocol address (receiver): 195.62.2.42

FRAME #11
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-08-02-8F-DA-6E
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 40 bytes
TTL: 128
Protocol: TCP
Header checksum: 7868
```

```
IP (sender): 195.62.2.42
IP (receiver): 205.188.9.82
Data: 08A714462F17E58D7E6632E45010FBD534AB0000

FRAME #12
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 40 bytes
TTL: 118
Protocol: TCP
Header checksum: 12680
IP (sender): 62.167.64.216
IP (receiver): 195.62.2.11
Data: 6EB7001912E6B4CEE83397325010FF9CB5830000

FRAME #13
MAC (receiver): FF-FF-FF-FF-FF-FF
MAC (sender): 00-50-8B-95-40-A8
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 78 bytes
TTL: 128
Protocol: UDP
Header checksum: 60155
IP (sender): 195.62.2.14
IP (receiver): 195.62.2.63
Data:
00890089003A4A82B6A2011000010000000000020454A454F4645454646434547
4542454445464341434143414341434143414341424D0000200001

FRAME #14
MAC (receiver): FF-FF-FF-FF-FF-FF
MAC (sender): 00-50-8B-95-40-A8
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 78 bytes
TTL: 128
Protocol: UDP
Header checksum: 59899
IP (sender): 195.62.2.14
IP (receiver): 195.62.2.63
Data:
00890089003A4B86B69E0110000100000000000020454A454F4645454646434547
4542454445464341434143414341434143414341424C0000200001
```

```
FRAME #15
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 202 bytes
TTL: 235
Protocol: UDP
Header checksum: 16966
IP (sender): 168.95.1.14
IP (receiver): 195.62.2.11
Data:
0035ADEB00B691A52070840000010001000200030238360237350331363503313138
3807696E2D6164647204617270610000C0001C00C000C0001000151800021 0D31
31382D3136352D37352D38360764796E616D69630568696E6574036E657400C012
000200010001518000090669706 46E7331C04EC012000200010001518000090669
70646E7332C04EC0650001000100011BF20004A85FC00EC07A0001000100011BFE
0004A85F010E000291000000000000000

FRAME #16
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 88 bytes
TTL: 255
Protocol: UDP
Header checksum: 3474
IP (sender): 195.62.2.11
IP (receiver): 168.95.192.14
Data:
ADEB00350044C1B88A6F0000000100000000000 10D3131382D3136352D37352D38
360764796E616D69630568696E6574036E6574000001000100002910000000000000
0000

FRAME #17
MAC (receiver): 01-80-C2-00-00-00
MAC (sender): 00-04-4D-8A-B0-D5
Protocol type: Ethernet 802.2 (LLC)
Data length: 38 bytes
Data:
0026030000000000800000044D8AB0C0000000008000000 44D8AB0C08023000014
0002000F00

FRAME #18
MAC (receiver): 00-90-27-A1-36-D0
MAC (sender): 00-02-16-09-FA-40
Frame type: Ethernet II (DIX)
Protocol type: IPv4
```

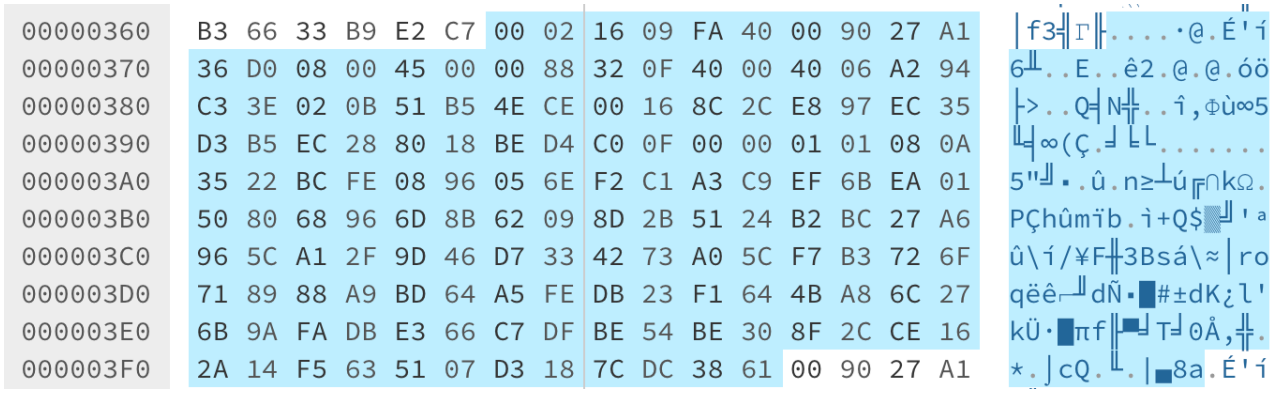
```
Header length: 20 bytes
Type of service: 0
Data length: 196 bytes
TTL: 51
Protocol: TCP
Header checksum: 29585
IP (sender): 81.181.78.206
IP (receiver): 195.62.2.11
Data:
88330016D3409239E86CFA2A8018003AC79C00000101080A0895FD0E3522B99400
00008C051E0000008100B328B791177C993A6F73BA71D2DCCE4E86CC9B37E5EF6D
30D82B6D0E6AD9B80A3FFCC9A2B32DC1634428DF6D0D7C5A0B3238F676D6DE3AD2
C262EE4C435399C273F6E376BCAB17921CDB968038A1EB1E190BAA368EEA7EF938
EEF029D7C789840A4D2062AC372771DC42B30C5922E4E62A0DC5FE883A87C83762
746CA10A945A0000000000

FRAME #19
MAC (receiver): 00-02-16-09-FA-40
MAC (sender): 00-90-27-A1-36-D0
Frame type: Ethernet II (DIX)
Protocol type: IPv4
Header length: 20 bytes
Type of service: 0
Data length: 76 bytes
TTL: 255
Protocol: UDP
Header checksum: 20435
IP (sender): 195.62.2.11
IP (receiver): 217.71.128.77
Data:
007B007B003831C11B0306F000001264000005DBD947804DCCA6B4FE31000000CC
A6B4FE30BAD000CCA6B4FE31000000CCA6B53E308F8000

-------------------------------------
Frames count: 19

[Types]
Ethernet 802.2 (LLC): 1
Ethernet 802.2 (SNAP): 0
Ethernet 802.3 (Raw/Novell): 0
Ethernet II (DIX): 18
    IPv4: 16
    ARP: 2
    RARP: 0
-------------------------------------
```

5. Полный анализ кадра

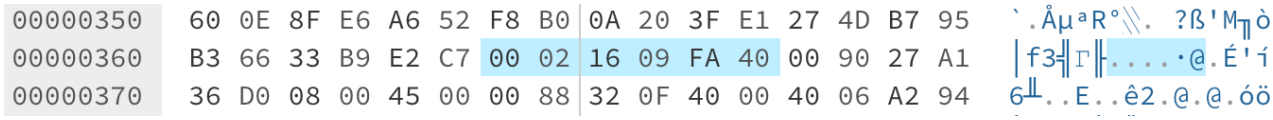
В рамках работы необходимо провести полный анализ кадра №9 из файла *ethers09.bin* при помощи шестнадцатеричного редактора:

Frame #9 из файла ethers09.bin

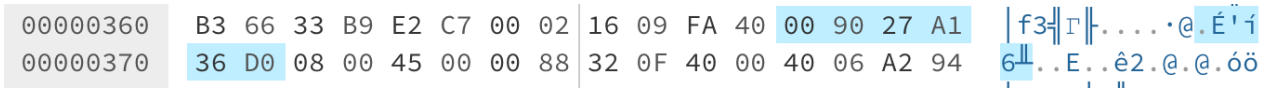
Данному кадру принадлежат байты с 870_{10} по 1020_{10} включительно, что соответствует индексам с 366_{16} по $3FC_{16}$:



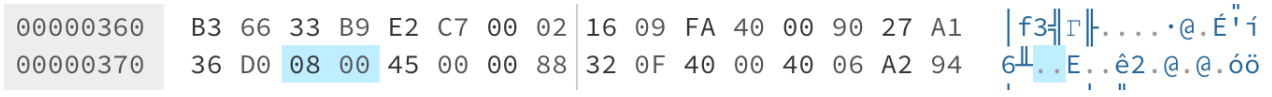
Первые 6 байт выделены для MAC-адреса получателя: 00-02-16-09FA-40:



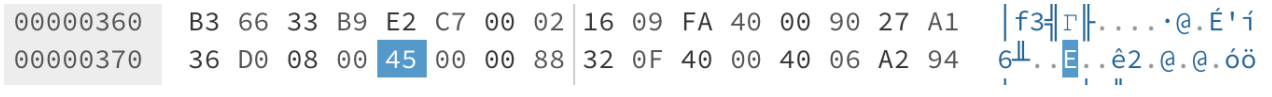
Следующие 6 байт являются MAC-адресом отправителя: 00-90-27-A1-36-D0:



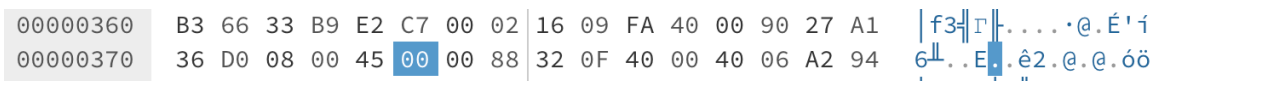
Далее идут два байта ответственные за длину данных или их тип. Так как записанное в них значение 0x0800 превышает 0x05DC (значение максимальной длины поля дейтаграммы), значит кадр Ethernet_II. Значение 0x0800 используют для IP-дейтаграмм.



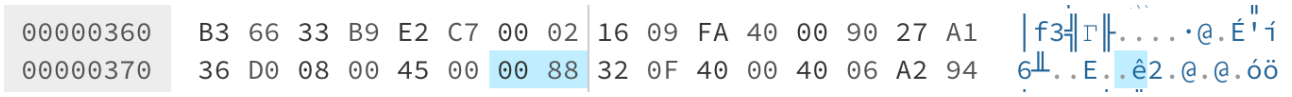
Один байт определяет версию протокола (4 бита) и длину заголовка (4 бита). В данном случае: версия “IPv4”, длина заголовка: 20 байт.



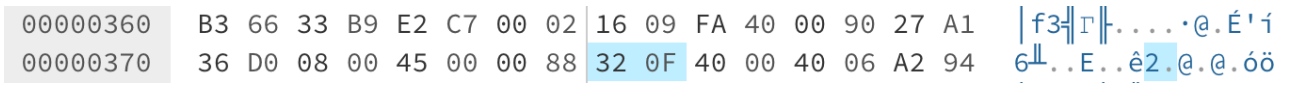
Ещё один байт определен для типа службы (ToS): 0



Далее два байта выделены под полную длину IP-дейтаграммы: 0x0088 = 136 байт

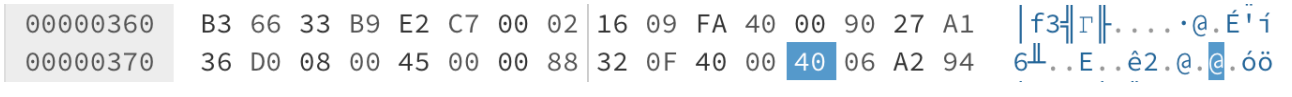


Здесь два байта являются 16-ти разрядным идентификатором:

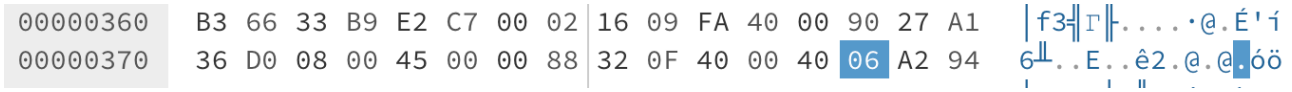


Ещё два байта выделены под флаги (3 бита) и 13-ти разрядное смещение объекта:

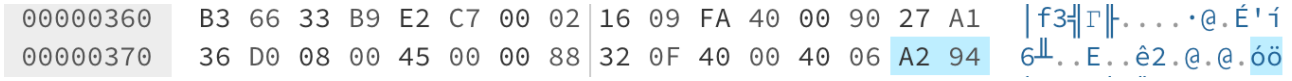
Далее один байт, определяющий время жизни (TTL): TTL = 64



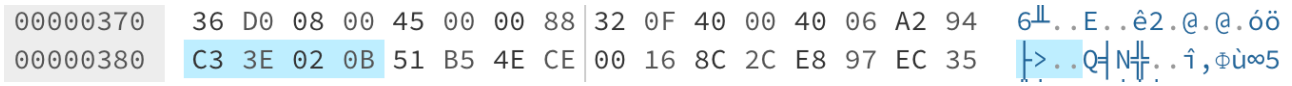
Ещё один байт под протокол транспортного уровня: 0x06 для TCP



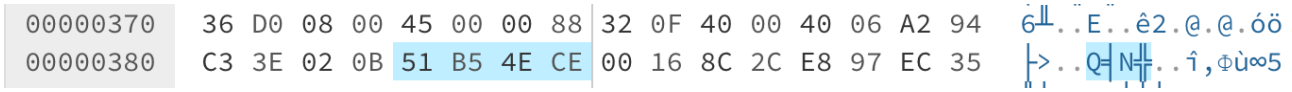
После идут два байта, в которые записывают контрольную сумму заголовка. Для данного кадра — 0xA294 = 41620



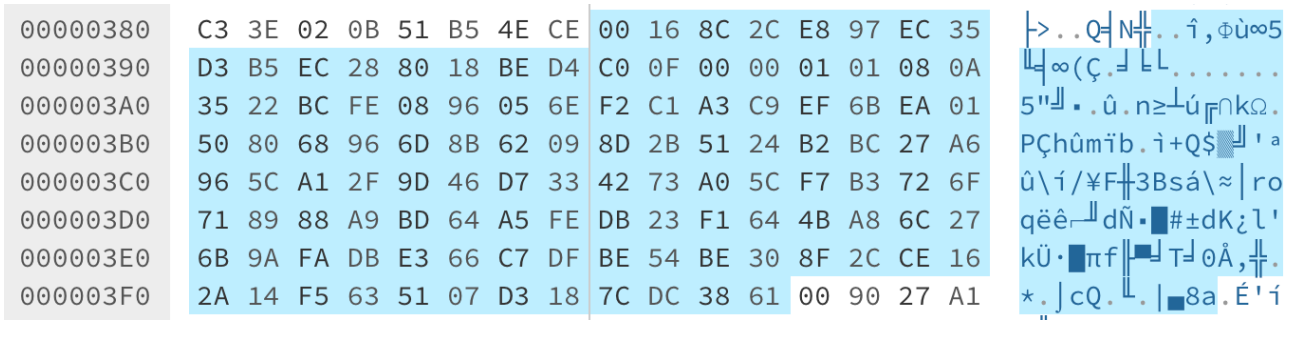
Дальше четыре байта под IP-адрес отправителя: 195.62.2.11



И четыре байта под IP-адрес получателя: 81.181.78.206

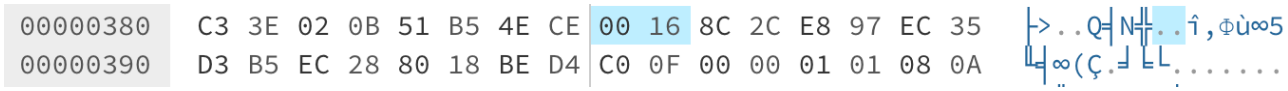


Оставшиеся 136 – 20 = 116 байт являются передаваемыми данными:



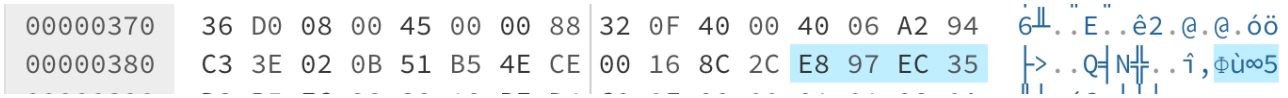
TCP-сегмент:

Два байта под порт отправителя: 0x0016 = 22

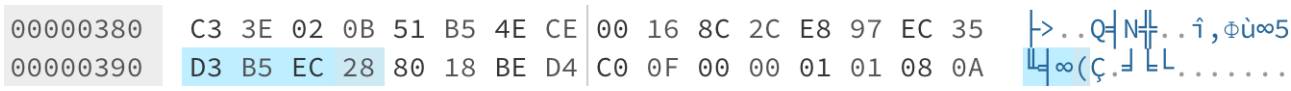


Два байта под порт получателя: 0x8C2C = 35884

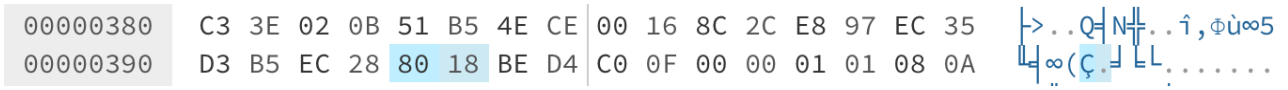
Четыре байта – код позиции в сообщении: E8 97 EC 35



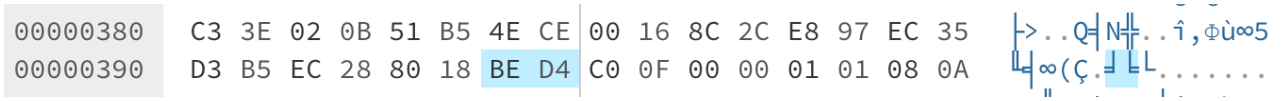
Ещё четыре байта – номер следующего октета: D3 B5 EC 28



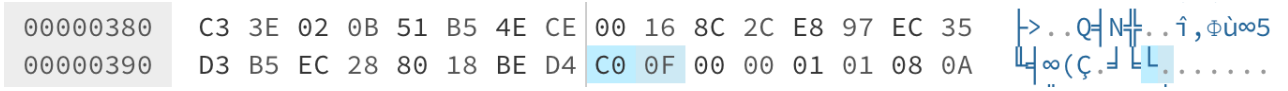
Далее идут два байта, из них – четыре бита, определяющих длину заголовка сегмента, затем шесть бит занимает резервное поле, оставшиеся шесть бит предназначены для флагов:



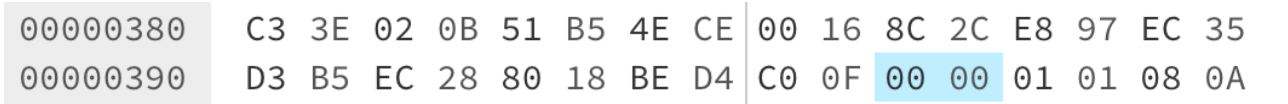
Два байта под размер окна (принимаемое количество октетов): 0xBED4 = 48824



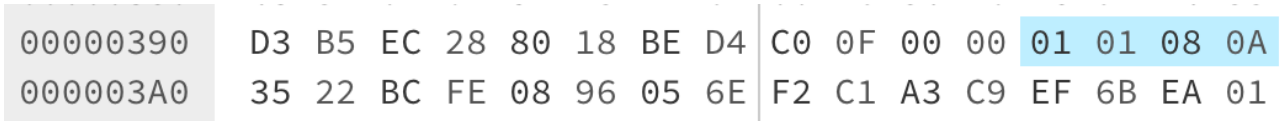
Ещё два байта – контрольная сумма:



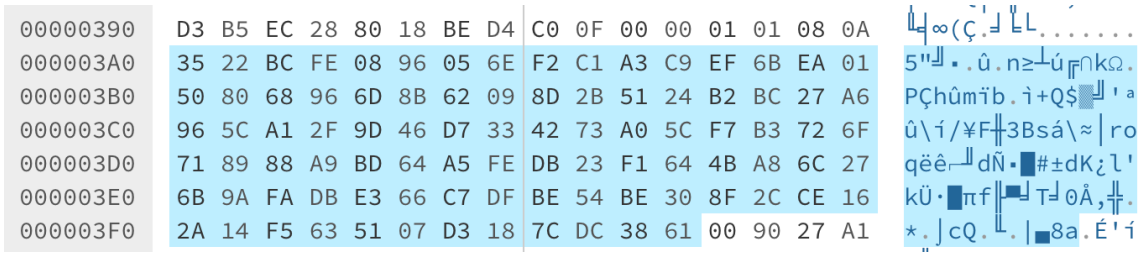
Следующие два байта – указатель важной информации: 0



Четыре байта выделяются под опции + заполнитель:



Оставшиеся байты (в данном случае 92 байта) содержат данные:



00000390	D3	B5	EC	28	80	18	BE	D4	C0	0F	00	00	01	01	08	0A
000003A0	35	22	BC	FE	08	96	05	6E	F2	C1	A3	C9	EF	6B	EA	01
000003B0	50	80	68	96	6D	8B	62	09	8D	2B	51	24	B2	BC	27	A6
000003C0	96	5C	A1	2F	9D	46	D7	33	42	73	A0	5C	F7	B3	72	6F
000003D0	71	89	88	A9	BD	64	A5	FE	DB	23	F1	64	4B	A8	6C	27
000003E0	6B	9A	FA	DB	E3	66	C7	DF	BE	54	BE	30	8F	2C	CE	16
000003F0	2A	14	F5	63	51	07	D3	18	7C	DC	38	61	00	90	27	A1

6. Вывод

В ходе лабораторной работы удалось спроектировать и реализовать программу, выполняющую анализ структуры кадра/фрейма технологии Ethernet. Ручной анализ фрейма подтвердил корректность работы программы, результаты совпали. Контрольные вопросы проработаны.