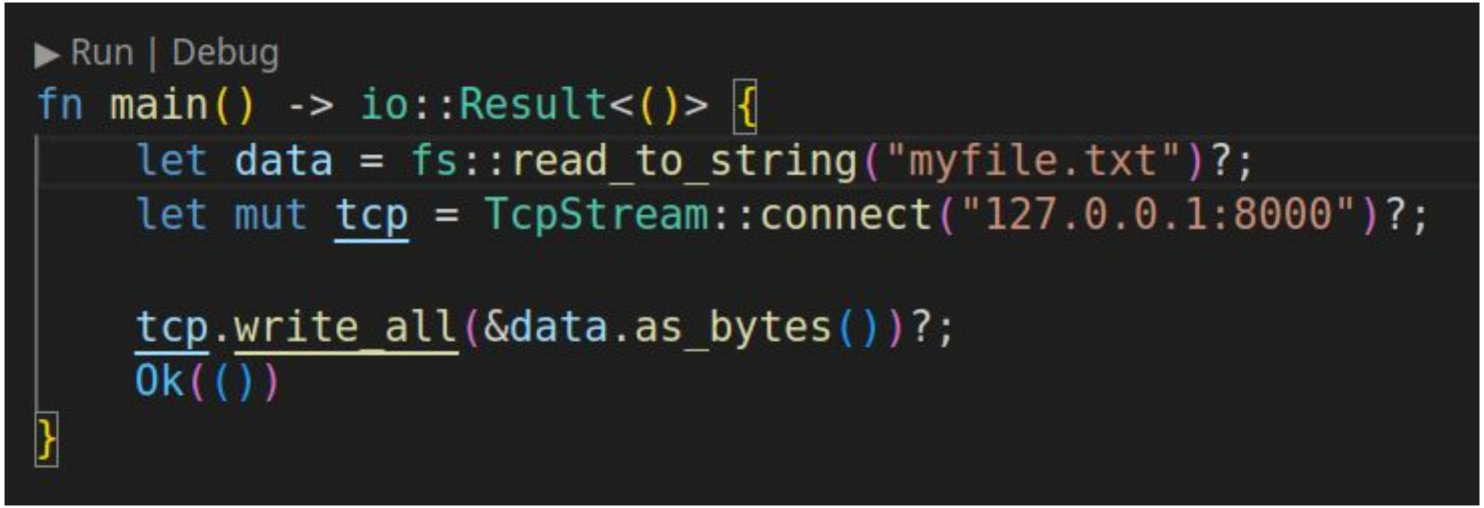
Мастерская №1 “Сообщение доставлено”

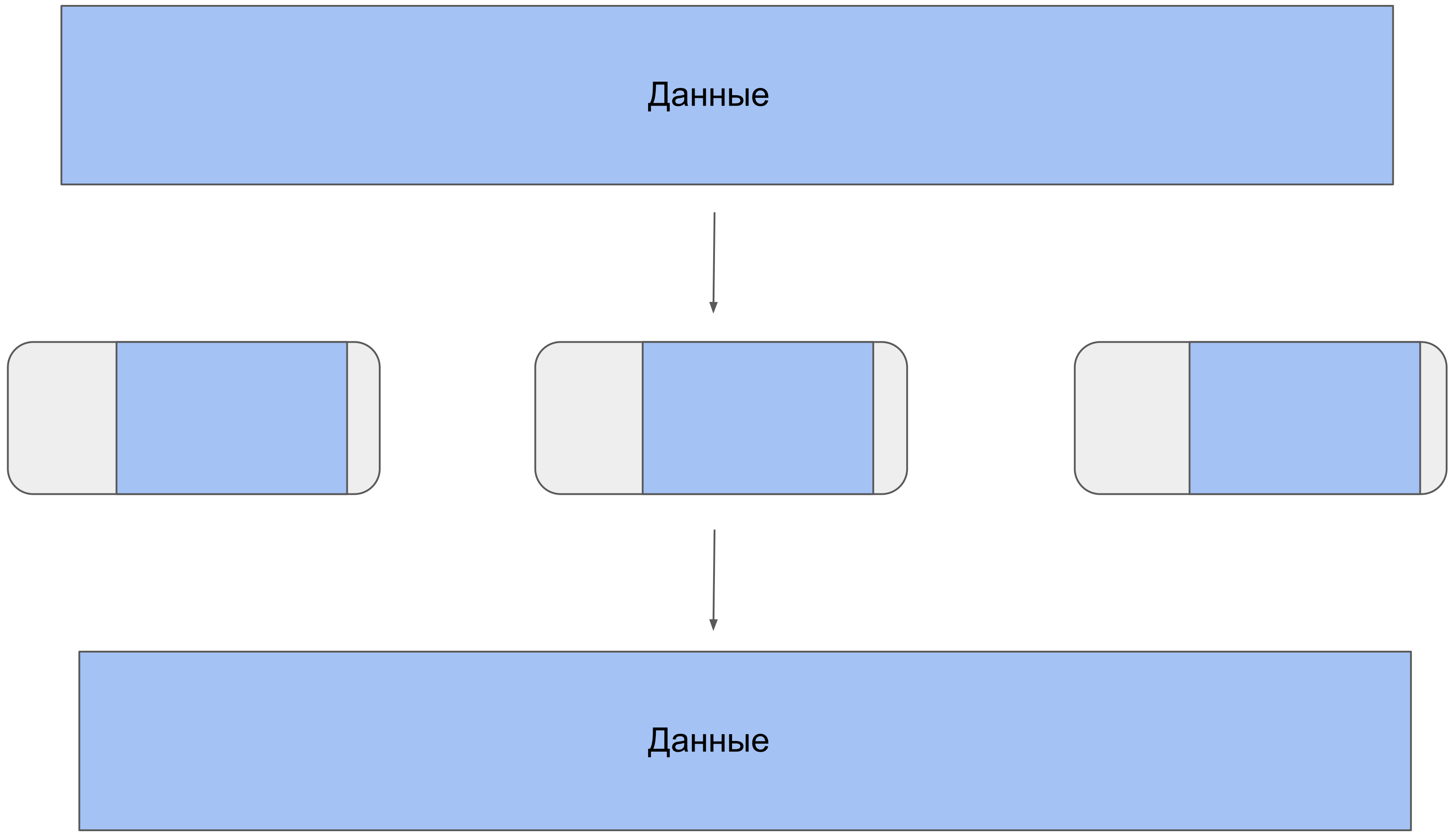
Гурчев Данил
Ярославцев Станислав
Самышкин Константин

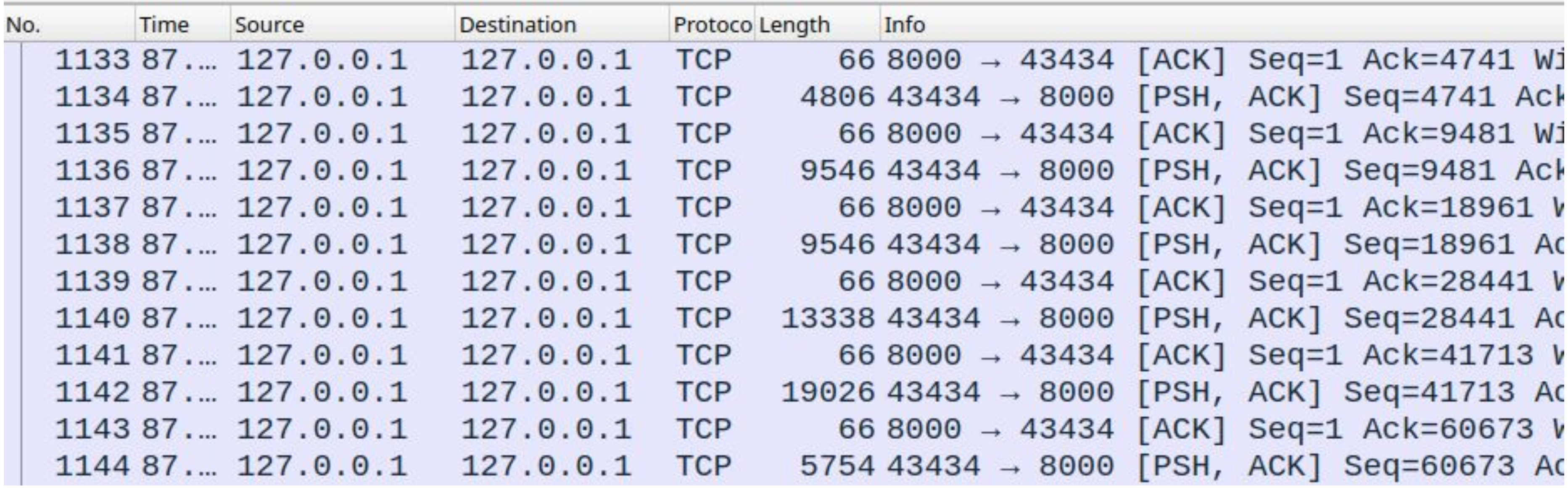
Куклин Георгий (подмастерье)
Климов Николай (мастер)

Как мы обычно отправляем данные по сети?



```rust
fn main() -> io::Result<()> {
    let data = fs::read_to_string("myfile.txt")?;
    let mut tcp = TcpStream::connect("127.0.0.1:8000")?;

    tcp.write_all(&data.as_bytes())?;
    Ok(())
}
```

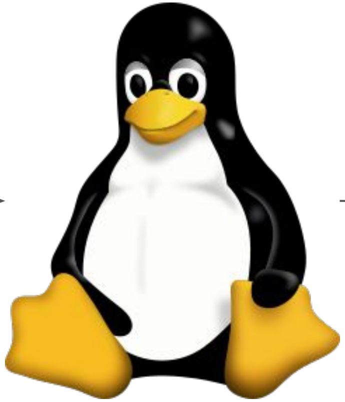

Данные
Данные



```
No.    Time   Source      Destination  Protoco Length Info
  1133 87.…   127.0.0.1   127.0.0.1    TCP         66 8000 → 43434 [ACK] Seq=1 Ack=4741 Wi
  1134 87.…   127.0.0.1   127.0.0.1    TCP       4806 43434 → 8000 [PSH, ACK] Seq=4741 Ack
  1135 87.…   127.0.0.1   127.0.0.1    TCP         66 8000 → 43434 [ACK] Seq=1 Ack=9481 Wi
  1136 87.…   127.0.0.1   127.0.0.1    TCP       9546 43434 → 8000 [PSH, ACK] Seq=9481 Ack
  1137 87.…   127.0.0.1   127.0.0.1    TCP         66 8000 → 43434 [ACK] Seq=1 Ack=18961 W
  1138 87.…   127.0.0.1   127.0.0.1    TCP       9546 43434 → 8000 [PSH, ACK] Seq=18961 Ac
  1139 87.…   127.0.0.1   127.0.0.1    TCP         66 8000 → 43434 [ACK] Seq=1 Ack=28441 W
  1140 87.…   127.0.0.1   127.0.0.1    TCP      13338 43434 → 8000 [PSH, ACK] Seq=28441 Ac
  1141 87.…   127.0.0.1   127.0.0.1    TCP         66 8000 → 43434 [ACK] Seq=1 Ack=41713 W
  1142 87.…   127.0.0.1   127.0.0.1    TCP      19026 43434 → 8000 [PSH, ACK] Seq=41713 Ac
  1143 87.…   127.0.0.1   127.0.0.1    TCP         66 8000 → 43434 [ACK] Seq=1 Ack=60673 W
  1144 87.…   127.0.0.1   127.0.0.1    TCP       5754 43434 → 8000 [PSH, ACK] Seq=60673 Ac
```

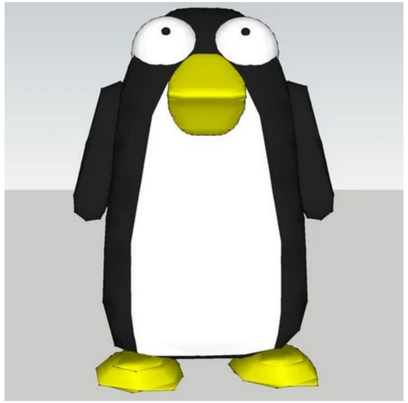
Кто это все делает?

tcp.write_all(...)

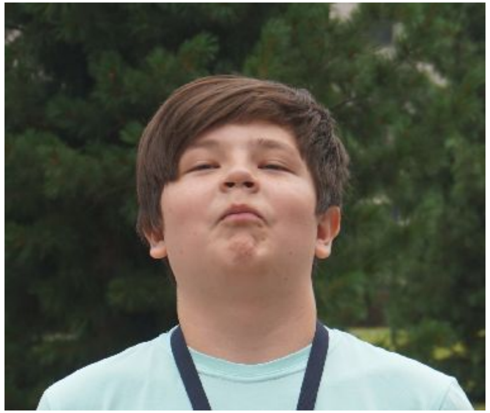


Как мы хотим?

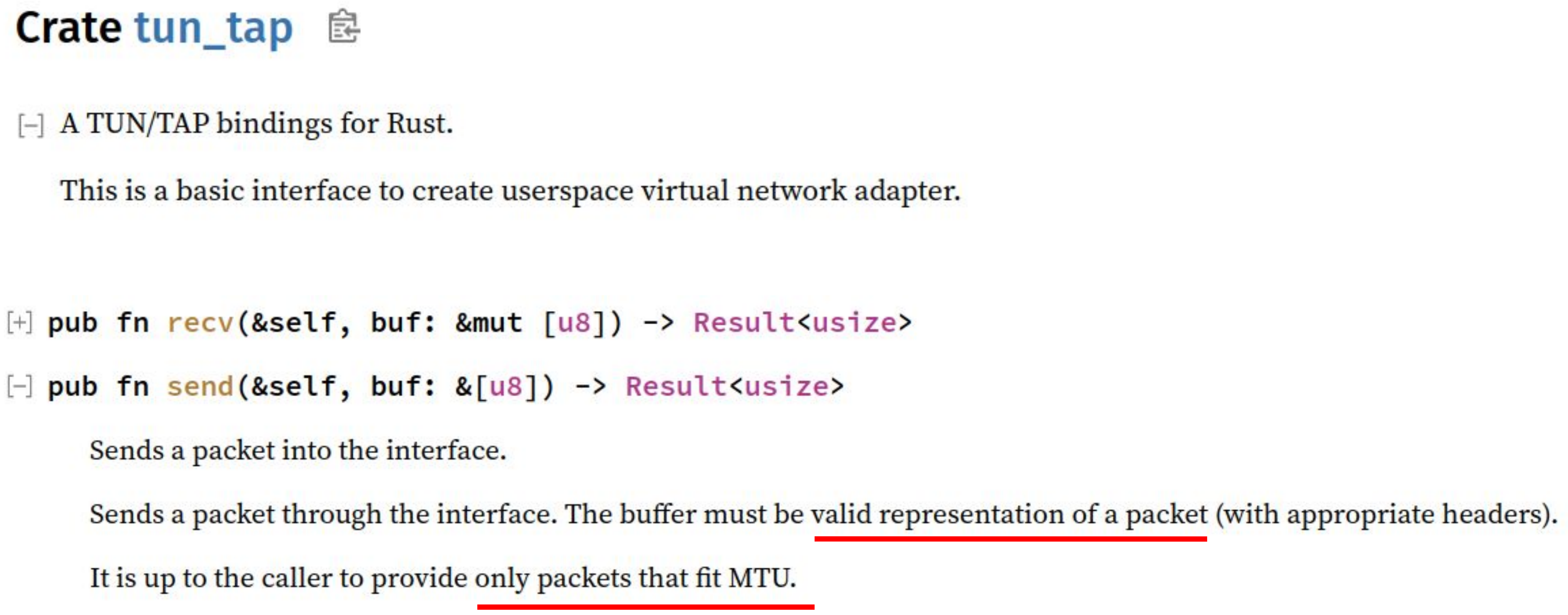
`tcp.write_all(...)` →



→



Какой у нас есть интерфейс



Crate tun_tap

[−] A TUN/TAP bindings for Rust.

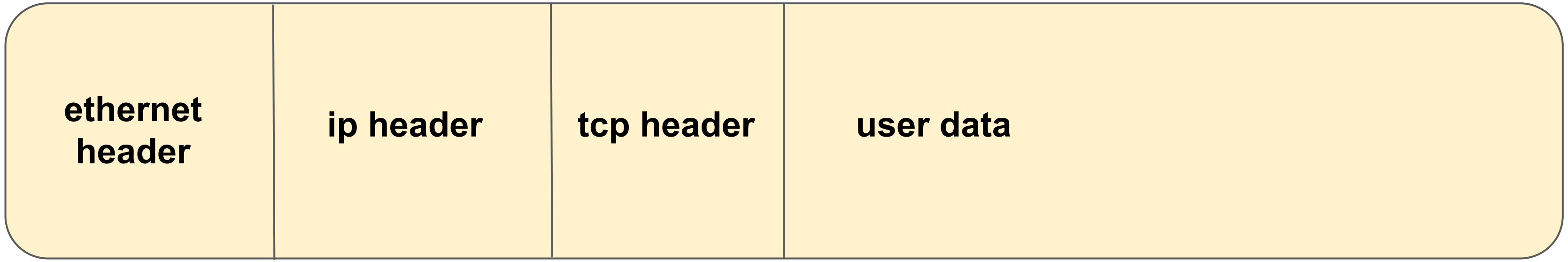
This is a basic interface to create userspace virtual network adapter.

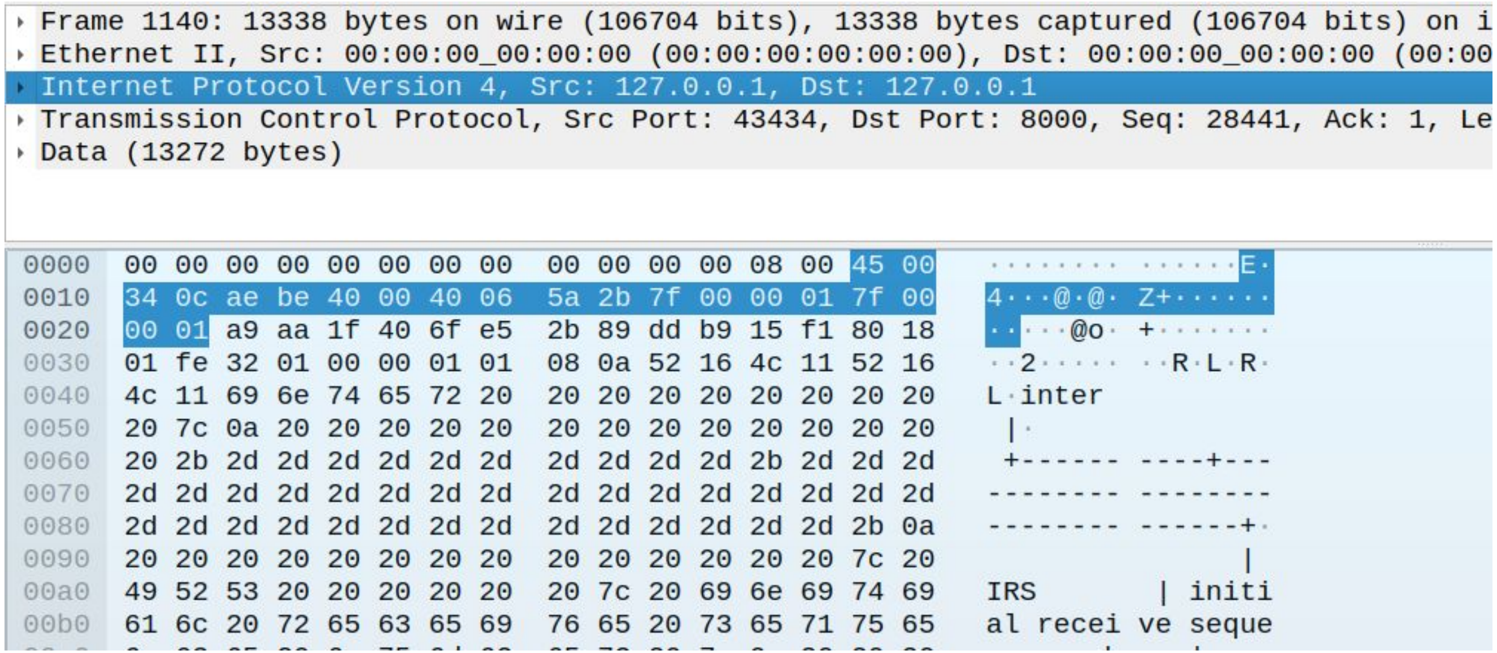
```
[+] pub fn recv(&self, buf: &mut [u8]) -> Result<usize>
[−] pub fn send(&self, buf: &[u8]) -> Result<usize>
```

Sends a packet into the interface.

Sends a packet through the interface. The buffer must be valid representation of a packet (with appropriate headers).

It is up to the caller to provide only packets that fit MTU.





```
▸ Frame 1140: 13338 bytes on wire (106704 bits), 13338 bytes captured (106704 bits) on i
▸ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 43434, Dst Port: 8000, Seq: 28441, Ack: 1, Le
▸ Data (13272 bytes)

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ········ ······E·
0010  34 0c ae be 40 00 40 06  5a 2b 7f 00 00 01 7f 00   4···@·@· Z+······
0020  00 01 a9 aa 1f 40 6f e5  2b 89 dd b9 15 f1 80 18   ·····@o· +·······
0030  01 fe 32 01 00 00 01 01  08 0a 52 16 4c 11 52 16   ··2····· ··R·L·R·
0040  4c 11 69 6e 74 65 72 20  20 20 20 20 20 20 20 20   L·inter
0050  20 7c 0a 20 20 20 20 20  20 20 20 20 20 20 20 20    |·
0060  20 2b 2d 2d 2d 2d 2d 2d  2d 2d 2d 2d 2b 2d 2d 2d    +------ ----+---
0070  2d 2d 2d 2d 2d 2d 2d 2d  2d 2d 2d 2d 2d 2d 2d 2d   -------- --------
0080  2d 2d 2d 2d 2d 2d 2d 2d  2d 2d 2d 2d 2d 2d 2b 0a   -------- ------+·
0090  20 20 20 20 20 20 20 20  20 20 20 20 20 20 7c 20                  |
00a0  49 52 53 20 20 20 20 20  20 7c 20 69 6e 69 74 69   IRS      | initi
00b0  61 6c 20 72 65 63 65 69  76 65 20 73 65 71 75 65   al recei ve seque
```

3. SPECIFICATION

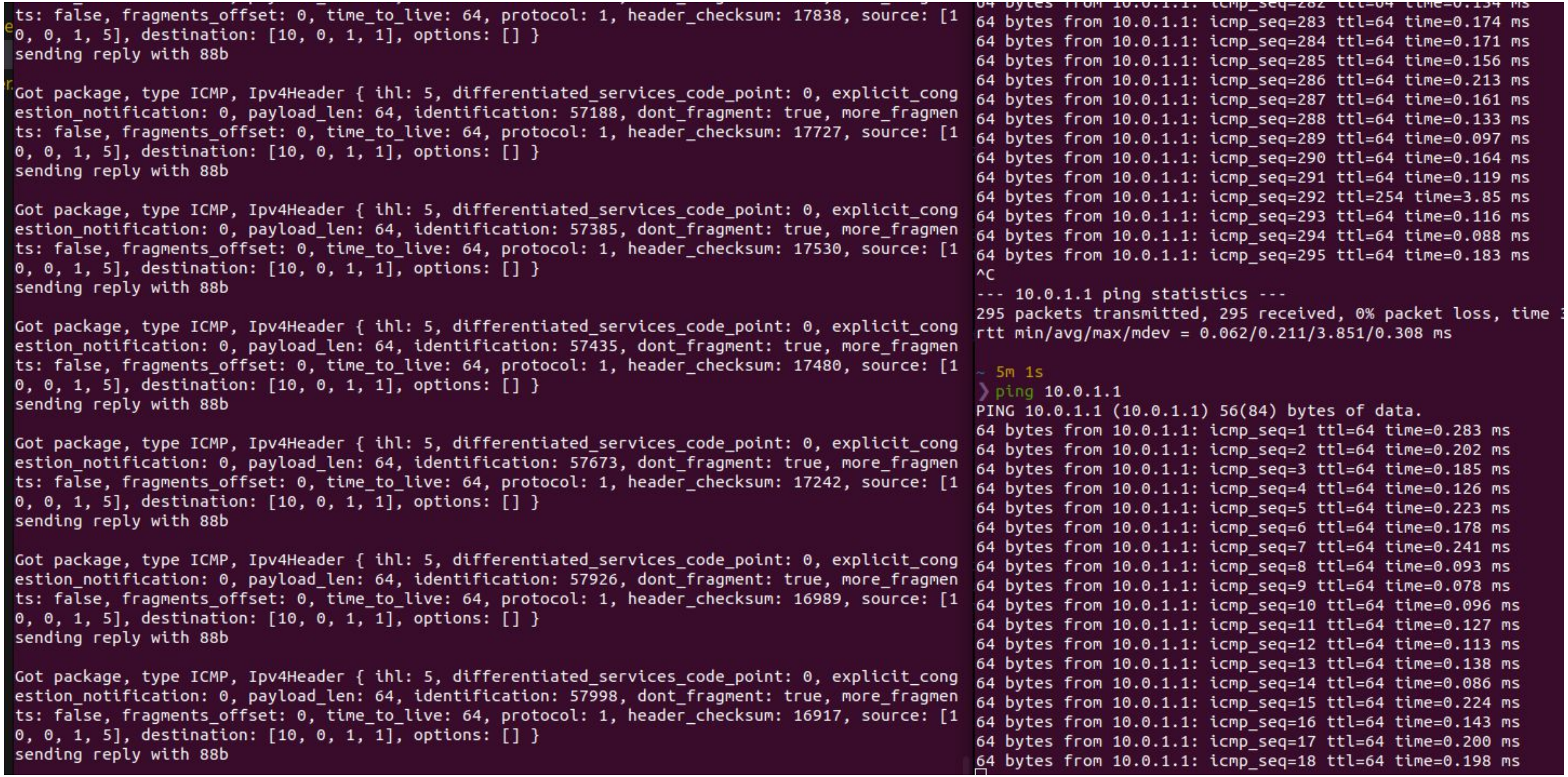
3.1. Internet Header Format

A summary of the contents of the internet header follows:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Version|  IHL  |Type of Service|          Total Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Identification        |Flags|      Fragment Offset    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Time to Live |    Protocol   |         Header Checksum       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                       Source Address                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Destination Address                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Options                    |    Padding    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Datagram Header

Figure 4.

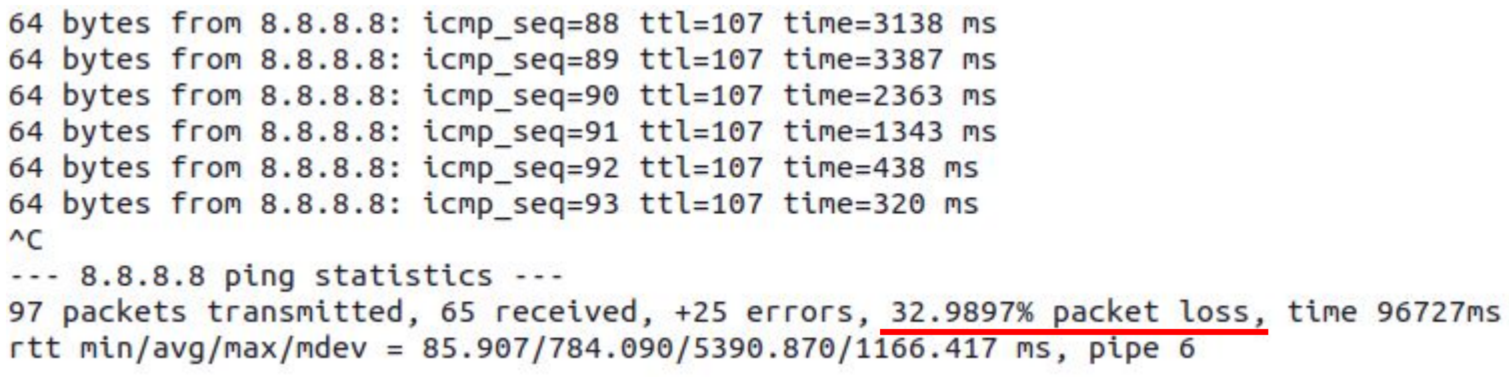


```
ts: false, fragments_offset: 0, time_to_live: 64, protocol: 1, header_checksum: 17838, source: [1
0, 0, 1, 5], destination: [10, 0, 1, 1], options: [] }
sending reply with 88b

Got package, type ICMP, Ipv4Header { ihl: 5, differentiated_services_code_point: 0, explicit_cong
estion_notification: 0, payload_len: 64, identification: 57188, dont_fragment: true, more_fragmen
ts: false, fragments_offset: 0, time_to_live: 64, protocol: 1, header_checksum: 17727, source: [1
0, 0, 1, 5], destination: [10, 0, 1, 1], options: [] }
sending reply with 88b

Got package, type ICMP, Ipv4Header { ihl: 5, differentiated_services_code_point: 0, explicit_cong
estion_notification: 0, payload_len: 64, identification: 57385, dont_fragment: true, more_fragmen
ts: false, fragments_offset: 0, time_to_live: 64, protocol: 1, header_checksum: 17530, source: [1
0, 0, 1, 5], destination: [10, 0, 1, 1], options: [] }
sending reply with 88b

Got package, type ICMP, Ipv4Header { ihl: 5, differentiated_services_code_point: 0, explicit_cong
estion_notification: 0, payload_len: 64, identification: 57435, dont_fragment: true, more_fragmen
ts: false, fragments_offset: 0, time_to_live: 64, protocol: 1, header_checksum: 17480, source: [1
0, 0, 1, 5], destination: [10, 0, 1, 1], options: [] }
sending reply with 88b

Got package, type ICMP, Ipv4Header { ihl: 5, differentiated_services_code_point: 0, explicit_cong
estion_notification: 0, payload_len: 64, identification: 57673, dont_fragment: true, more_fragmen
ts: false, fragments_offset: 0, time_to_live: 64, protocol: 1, header_checksum: 17242, source: [1
0, 0, 1, 5], destination: [10, 0, 1, 1], options: [] }
sending reply with 88b

Got package, type ICMP, Ipv4Header { ihl: 5, differentiated_services_code_point: 0, explicit_cong
estion_notification: 0, payload_len: 64, identification: 57926, dont_fragment: true, more_fragmen
ts: false, fragments_offset: 0, time_to_live: 64, protocol: 1, header_checksum: 16989, source: [1
0, 0, 1, 5], destination: [10, 0, 1, 1], options: [] }
sending reply with 88b

Got package, type ICMP, Ipv4Header { ihl: 5, differentiated_services_code_point: 0, explicit_cong
estion_notification: 0, payload_len: 64, identification: 57998, dont_fragment: true, more_fragmen
ts: false, fragments_offset: 0, time_to_live: 64, protocol: 1, header_checksum: 16917, source: [1
0, 0, 1, 5], destination: [10, 0, 1, 1], options: [] }
sending reply with 88b
```

```
64 bytes from 10.0.1.1: icmp_seq=283 ttl=64 time=0.174 ms
64 bytes from 10.0.1.1: icmp_seq=284 ttl=64 time=0.171 ms
64 bytes from 10.0.1.1: icmp_seq=285 ttl=64 time=0.156 ms
64 bytes from 10.0.1.1: icmp_seq=286 ttl=64 time=0.213 ms
64 bytes from 10.0.1.1: icmp_seq=287 ttl=64 time=0.161 ms
64 bytes from 10.0.1.1: icmp_seq=288 ttl=64 time=0.133 ms
64 bytes from 10.0.1.1: icmp_seq=289 ttl=64 time=0.097 ms
64 bytes from 10.0.1.1: icmp_seq=290 ttl=64 time=0.164 ms
64 bytes from 10.0.1.1: icmp_seq=291 ttl=64 time=0.119 ms
64 bytes from 10.0.1.1: icmp_seq=292 ttl=254 time=3.85 ms
64 bytes from 10.0.1.1: icmp_seq=293 ttl=64 time=0.116 ms
64 bytes from 10.0.1.1: icmp_seq=294 ttl=64 time=0.088 ms
64 bytes from 10.0.1.1: icmp_seq=295 ttl=64 time=0.183 ms
^C
--- 10.0.1.1 ping statistics ---
295 packets transmitted, 295 received, 0% packet loss, time
rtt min/avg/max/mdev = 0.062/0.211/3.851/0.308 ms

~ 5m 1s
❯ ping 10.0.1.1
PING 10.0.1.1 (10.0.1.1) 56(84) bytes of data.
64 bytes from 10.0.1.1: icmp_seq=1 ttl=64 time=0.283 ms
64 bytes from 10.0.1.1: icmp_seq=2 ttl=64 time=0.202 ms
64 bytes from 10.0.1.1: icmp_seq=3 ttl=64 time=0.185 ms
64 bytes from 10.0.1.1: icmp_seq=4 ttl=64 time=0.126 ms
64 bytes from 10.0.1.1: icmp_seq=5 ttl=64 time=0.223 ms
64 bytes from 10.0.1.1: icmp_seq=6 ttl=64 time=0.178 ms
64 bytes from 10.0.1.1: icmp_seq=7 ttl=64 time=0.241 ms
64 bytes from 10.0.1.1: icmp_seq=8 ttl=64 time=0.093 ms
64 bytes from 10.0.1.1: icmp_seq=9 ttl=64 time=0.078 ms
64 bytes from 10.0.1.1: icmp_seq=10 ttl=64 time=0.096 ms
64 bytes from 10.0.1.1: icmp_seq=11 ttl=64 time=0.127 ms
64 bytes from 10.0.1.1: icmp_seq=12 ttl=64 time=0.113 ms
64 bytes from 10.0.1.1: icmp_seq=13 ttl=64 time=0.138 ms
64 bytes from 10.0.1.1: icmp_seq=14 ttl=64 time=0.086 ms
64 bytes from 10.0.1.1: icmp_seq=15 ttl=64 time=0.224 ms
64 bytes from 10.0.1.1: icmp_seq=16 ttl=64 time=0.143 ms
64 bytes from 10.0.1.1: icmp_seq=17 ttl=64 time=0.200 ms
64 bytes from 10.0.1.1: icmp_seq=18 ttl=64 time=0.198 ms
```

Проблемы передачи данных по сети

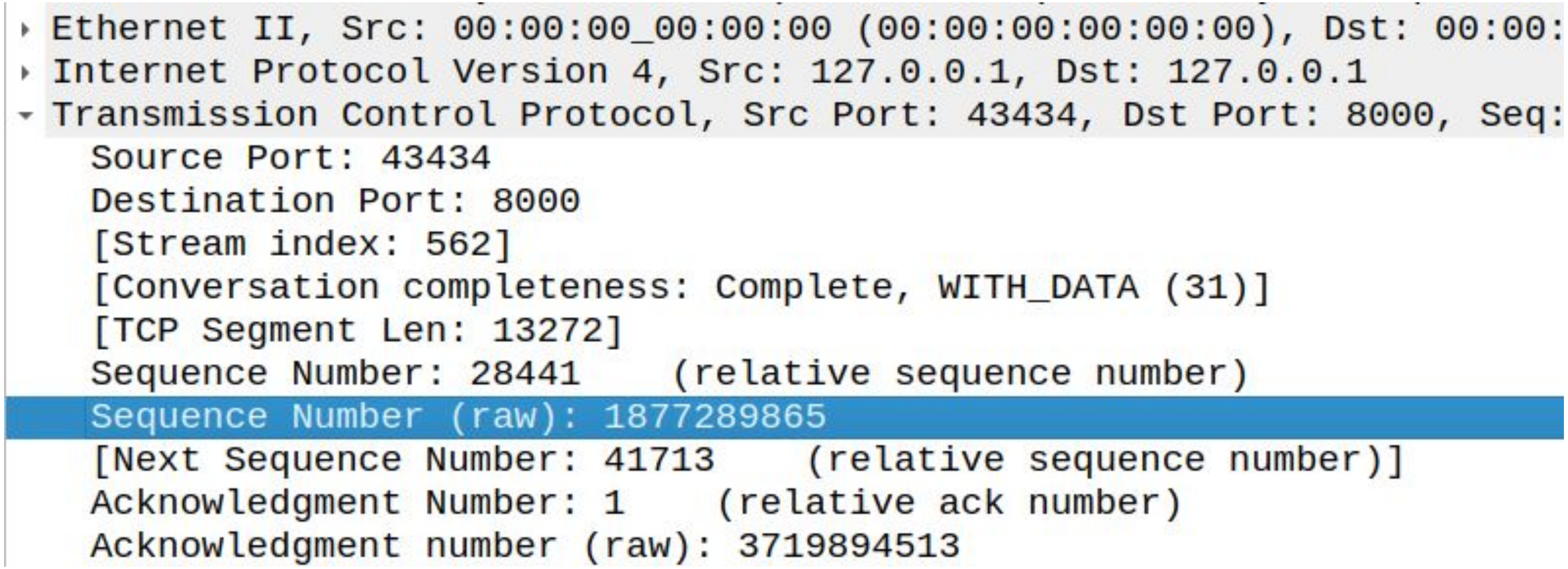
Пакеты могут повредиться, поменять порядок или вообще потеряться.



```
64 bytes from 8.8.8.8: icmp_seq=88 ttl=107 time=3138 ms
64 bytes from 8.8.8.8: icmp_seq=89 ttl=107 time=3387 ms
64 bytes from 8.8.8.8: icmp_seq=90 ttl=107 time=2363 ms
64 bytes from 8.8.8.8: icmp_seq=91 ttl=107 time=1343 ms
64 bytes from 8.8.8.8: icmp_seq=92 ttl=107 time=438 ms
64 bytes from 8.8.8.8: icmp_seq=93 ttl=107 time=320 ms
^C
--- 8.8.8.8 ping statistics ---
97 packets transmitted, 65 received, +25 errors, 32.9897% packet loss, time 96727ms
rtt min/avg/max/mdev = 85.907/784.090/5390.870/1166.417 ms, pipe 6
```

Решение - протокол TCP

Sequence number гарантирует доставку пакетов в правильном порядке.

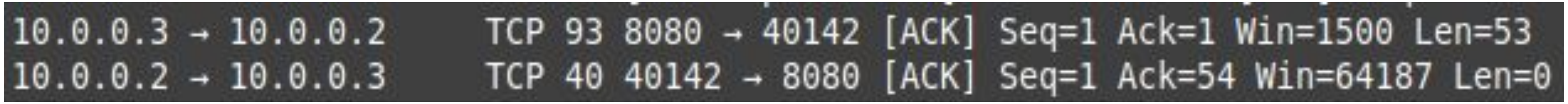


```
▸ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▾ Transmission Control Protocol, Src Port: 43434, Dst Port: 8000, Seq:
    Source Port: 43434
    Destination Port: 8000
    [Stream index: 562]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 13272]
    Sequence Number: 28441    (relative sequence number)
    Sequence Number (raw): 1877289865
    [Next Sequence Number: 41713    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 3719894513
```

Протокол TCP

Специальный пакет ACK отправляется в ответ на любые данные.

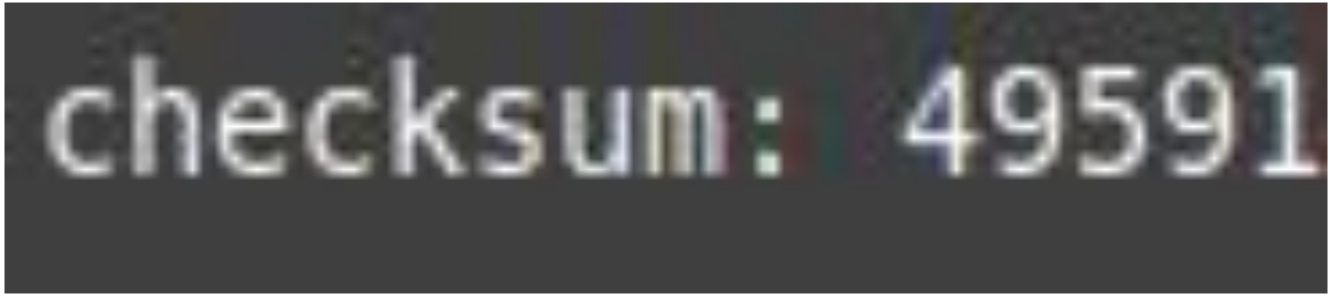
Это гарантирует доставку пакетов.



```
10.0.0.3 → 10.0.0.2     TCP 93 8080 → 40142 [ACK] Seq=1 Ack=1 Win=1500 Len=53
10.0.0.2 → 10.0.0.3     TCP 40 40142 → 8080 [ACK] Seq=1 Ack=54 Win=64187 Len=0
```

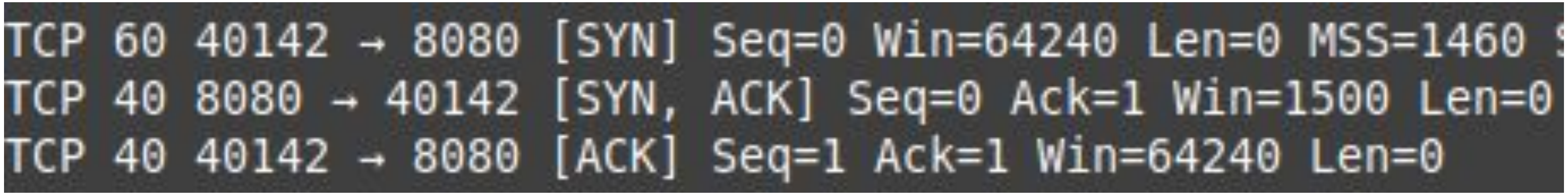
Протокол TCP

Checksum гарантирует целостность данных.



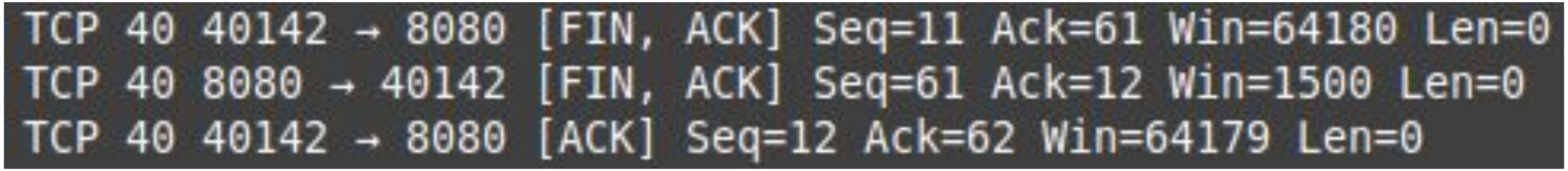
```
checksum: 49591
```

Протокол TCP: подключение (**3-Way Handshake**)



```
TCP 60 40142 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
TCP 40 8080 → 40142 [SYN, ACK] Seq=0 Ack=1 Win=1500 Len=0
TCP 40 40142 → 8080 [ACK] Seq=1 Ack=1 Win=64240 Len=0
```

Протокол TCP: разрыв подключения



```
TCP 40 40142 → 8080 [FIN, ACK] Seq=11 Ack=61 Win=64180 Len=0
TCP 40 8080 → 40142 [FIN, ACK] Seq=61 Ack=12 Win=1500 Len=0
TCP 40 40142 → 8080 [ACK] Seq=12 Ack=62 Win=64179 Len=0
```

Что не успели сделать

- retransmission (не очень хорошая версия)
- udp
- полноценное тестирование

FIN