CompSci-230: Homework 2

Jitao Zhang

1. Programs(Written in Nodejs)

Server side(in Google Cloud):

// Import builtin net module.

**var** net = require('net');

// Create and return a net.Server object

**var** server = net.createServer(**function**(socket) {

console.**log**('. client remote address : ' + socket.remoteAddress + ':' + socket.remotePort);

// set encoding type

socket.setEncoding('utf-8');

// When receive client data.

socket.on('data', **function** (data) {

// Print received client data

console.**log**('Receive client send data : ' + data);

// Server send data back to client use client net.Socket object.

socket.end('Server received data : ' + data);

});

// When client send data complete.

socket.on('end', **function** () {

console.**log**('Client disconnect.');

});

// When client timeout.

socket.on('timeout', **function** () {

console.**log**('Client request time out. ');

})

});

// Make the server a TCP server listening on port 5000.

server.listen(5000, **function** () {

// Get server address info.

**var** serverInfo = server.address();

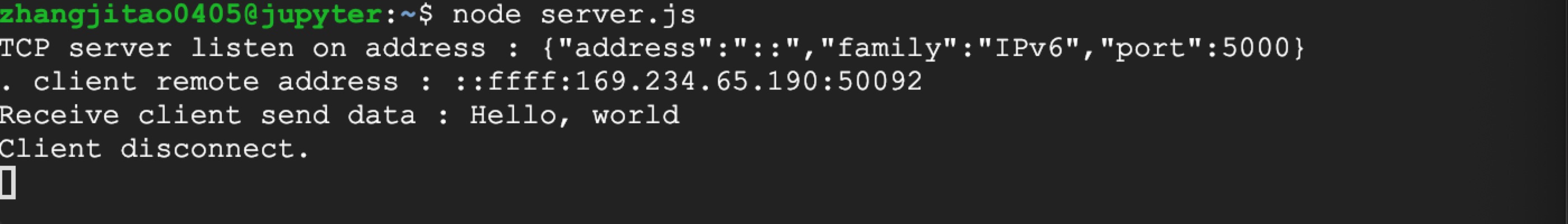
console.**log**('TCP server listen on address : ' + JSON.**stringify**(serverInfo));

server.on('close', **function** () {

console.**log**('TCP server is closed.');

});

});



Client Side(localhost):

// Import buildin net module.

**var** net = require('net');

**var** option = {

host:'35.243.136.73',

port: 5000

}

// Create TCP client.

**var** client = net.createConnection(option, **function** () {

console.**log**('Connection remote address : ' + client.remoteAddress + ":" + client.remotePort);

});

// set the encoding type

client.setEncoding('utf8');

// When receive server send back data.

client.on('data', **function** (data) {

console.**log**('Server return data is : ' + data);

});

// When connection disconnected.

client.on('end',**function** () {

console.**log**('Client socket disconnect. ');

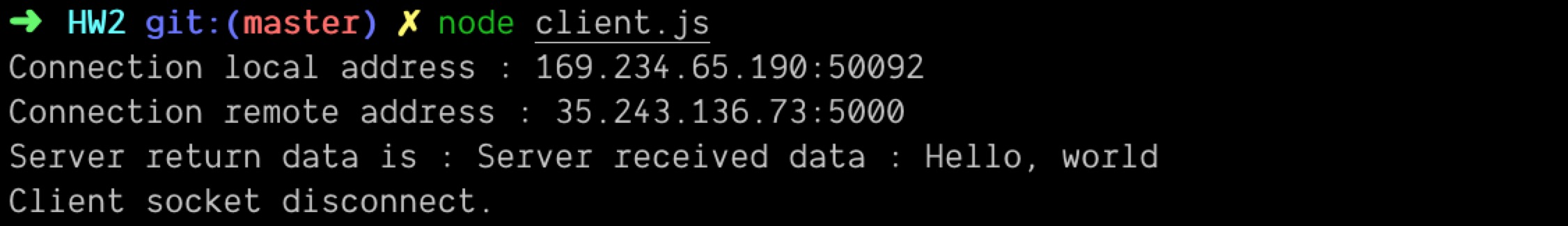
});

client.on('error', **function** (err) {

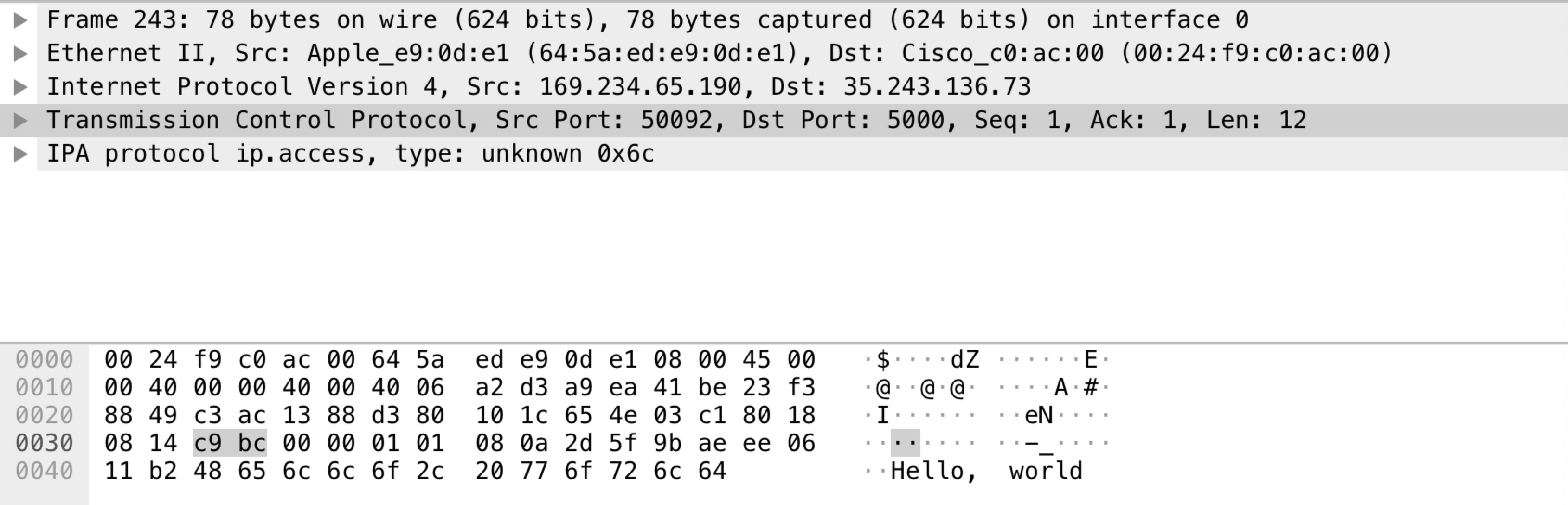
console.error(JSON.**stringify**(err));

});

client.write('Hello, world');

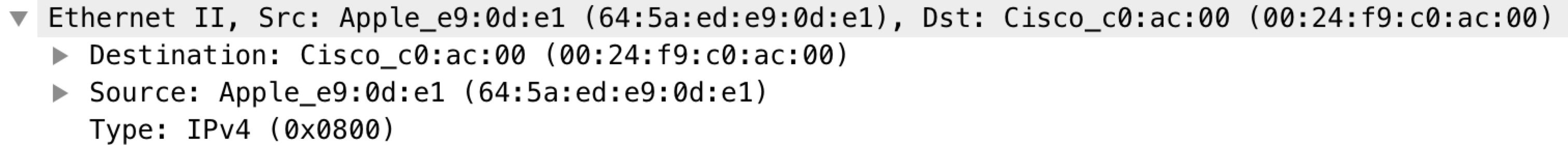


1. Packet from sender



1. Questions

* What is encoded in bytes 0-5 and 6-11?

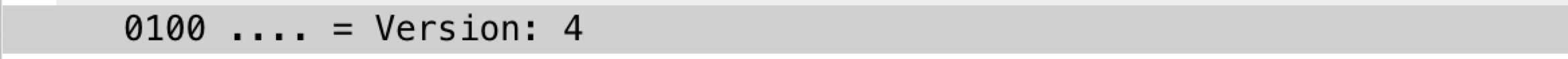


0-5 bytes => Destination MAC address: 00:24:f9:c0:ac:00

6-11 bytes => Source MAC address: 64:5a:ed:e9:0d:e1

* What is encoded in, and what is the relationship between, byte 14 and the two bytes  16-17?

byte 14 is showing IP version4



bytes 16 – 17 shows the total length is 64



The relationship is that the IP packet maximum length is 64k bytes

* What is encoded in bytes 18-19?



because the data might be split up into many packets, the identification is to maintain the order of it. The 0 here means it is the first packet.

* What is encoded in bytes 20-21?

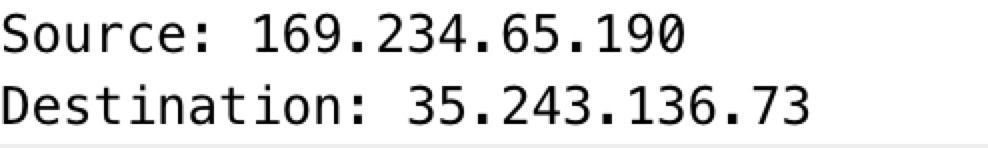


means this packet is not generated from fragment

* What is encoded in byte 23?



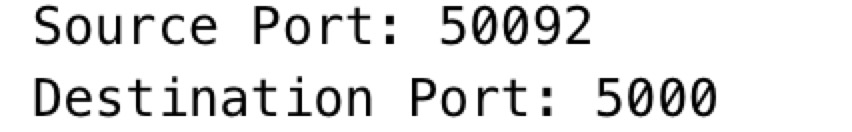
* What is encoded in bytes 26-29 and 30-33?



26 – 29 source IP: 169.234.65.190

30 – 33 destination IP: 35.243.136.73

* What is encoded in bytes 34-35 and 36-37?



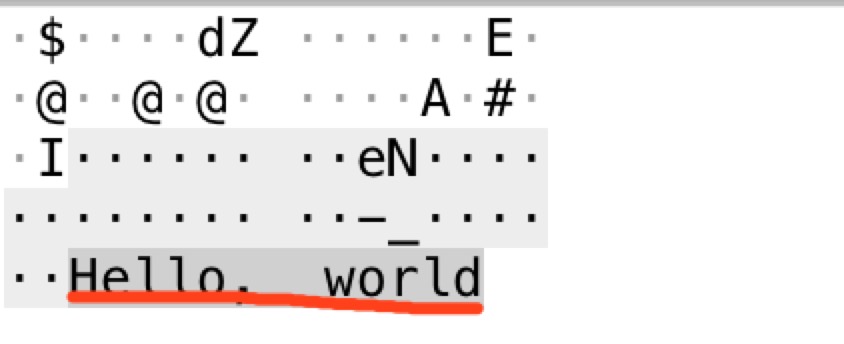
34 – 35 source port 50092

36 – 37 Destination port 5000

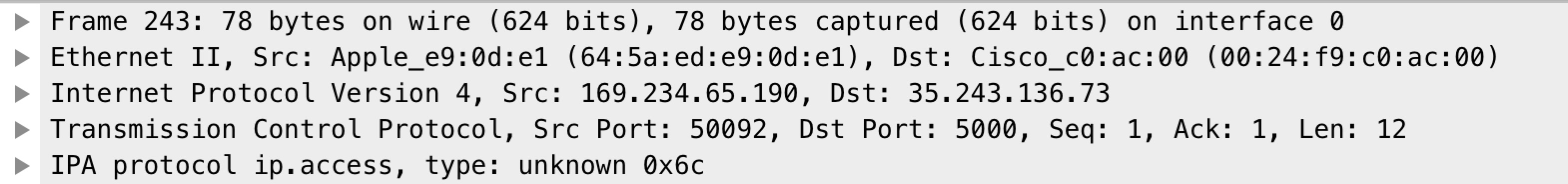
* What is encoded after byte 65?

The TCP payload data





* Finally, split the raw dump of bits associating each block of bits to the layers 2 (Link), 3 (Network), 4 (Transport), and 7 (Application).



78 bytes total:

0 – 13 link layer

14 – 33 network layer

34 – 65 transport layer

66 – 77 application layer