FEUP – Computer Networks 2021/2022

2.º Lab Assignment

Diogo Costa
up201906731@edu.fe.up.pt

Francisco Colino
up201905405@edu.fe.up.pt

January 24, 2022

Summary

This report describes the steps taken to configure a computer network with 2 VLANs using 3 computers, 1 switch, 1 Cisco router and connection to the internet to download a file with a download application also implemented and described in this report. The download application, which is a FTP client, is described first in detail. Regex was used to parse the URL given as argument and it uses sockets to create 2 connections (data and control connections). Afterwards, the network configuration and analysis is described for each of the four experiments.

1 Introduction

The objective of this assignment is to implement an FTP client (download application) and configure a computer network with 2 VLANs, composed by 3 computers, 1 switch, 1 cisco router and connection to the internet so that the application can be tested in it afterwards.

In the first part of this report we describe the download application implemented and in the second part the network configuration.

2 Part 1 – Download application

As part of this 2nd lab assignment, we had to implement a download application that uses the FTP (File Transfer Protocol), described in RFC959, and takes an an argument that adopts the URL syntax as described in RFC1738. In this specific case, the URl is in the following format:

$$ftp://[<user>:<password>@]<host>/<url-path>$$

2.1 Architecture of the download application

The application has two main modules: parser and FTP client. The parser module takes the argument given to the application and breaks it into *user*, *password*, *host*, and *url-path*. This is accomplished using the following regular expression defined in *parser.c*:

```
char *REGULAR_EXPRESSION = "ftp://((.+):(.+)@)?([^/@:]+)/([^\f\n\r\t\v
    \x20]*)";
```

The function that parses the input is:

```
parsed_params_t* parse_input_params(const char* input);
```

and the return structure is declared as follows in *parser.h*:

```
typedef struct parsed_params {
    char *user;
    char *password;
    char *host;
    char *url_path;
} parsed_params_t;
```

After the URL has been parsed into the structure, the FTP client can proceed.

First, we need to get the IP address associated with the given *host*. That is accomplished in the following function defined in *ftp.c*:

```
static char *get_ip(char *host_name);
```

The application now creates a socket and connects it to the obtained IP in the previous step in the FTP control port (21). Then, it uses the *user* and *password* to login. The function that does that is defined in *ftp.c*:

```
int ftp_login(int socket_fd, char *user, char *pass);
```

It sends a USER command with the given *user* and reads the response. If it has a 331 reply code (User name okay, need password.) it proceeds to send a PASS command with the given *password* and check if the response has a 230 reply code (User logged in, proceed.) which indicates that the user is logged in and can now proceed to the download.

The download is done in the following function defined in *ftp.c*:

```
int ftp_download_file(int socket_fd, char *host, char *path);
```

it sends a PASV command and gets the *port* in the response so that it can connect to that port to receive the file. Then, it sends a RETR command with the given *url-path* to the control connection and reads the file in the data connection. Once the download is finished it checks the reply codes to make sure the download occurred without errors.
Finally, the application sends a QUIT command in control connection to finish the connection.

2.2 Report of a successful download

The application was tested in several FTP servers both with username and as anonymous. The following image describes a usage in *netlab1.fe.up.pt*:

```
./download ftp://rcom:rcom@netlab1.fe.up.pt/files/pic1.jpg
USER: "rcom"
PASSWORD: "rcom"
HOST: "netlab1.fe.up.pt"
URL-PATH: "files/pic1.jpg"

RECV: 220 Welcome to netlab-FTP server
SENT: USER rcom
RECV: 331 Please specify the password.
SENT: PASS rcom
RECV: 230 Login successful.
SENT: PASV
RECV: 227 Entering Passive Mode (192,168,109,136,170,28).
SENT: RETR files/pic1.jpg
RECV: 150 Opening BINARY mode data connection for files/pic1.jpg (340603 bytes)
.
RECV: 226 Transfer complete.
SENT: QUIT
RECV: 221 Goodbye.
```

Figure 1: Successful download.

3 Part 2 – Network configuration and analysis

3.1 Experiment 1

3.1.1 Objectives

This experiment aims to configure the IP addresses of two computers and connect them to a Switch which should result in a network allowing both computers to communicate.

3.1.2 Network architecture

This experiment uses two *tux* computers (tux33, tux34) and the *Cisco* Switch. Each computer should be connected to the Switch using an *eth* port, for now both computers will use eth0.

3.1.3 Main configuration commands

First we need to configure the eth0 Network Interfaces on both computers, this is done using the following commands:

```
1 ifconfig eth0 up                  # Activates eth0 on both tuxs
2 ifconfig eth 0 172.16.30.1/24     # On tux33, configures it's ip address
      on eth0
3 ifconfig eth 0 172.16.30.254/24 # On tux34, configures it's ip address
      on eth0
4
5 route -n     # Inspect the routes that were setup
6 arp -a       # Check the arp tables
```

3.1.4 Analysis of the logs

ARP packets are used to map an IP address to a physical address (MAC). When a host wants to send a packet to another host, whose IP address is known but not the MAC, on the same LAN it first Broadcasts an ARP packet that asks for the MAC address associated with the destination's IP address. This is needed as Network

Interface Controllers don't have IP addresses but MAC addresses. We should now be able to ping the tux33 from tux34 and vice-versa.

The ping command uses ICMP packets, these packets are sent from both origin (request) and the destination (response). ICMP packets contain the Ether layer, which has both target and source's MAC addresses, and also contains the IP layer that holds the source's and target's IP addresses.

Ethernet frames have an header, and this header has an EtherType field, this field is what indicates which protocol is encased in the payload (ARP, IP, ICMP, etc.), the size of these frames can be determined by detecting the end of the frame that is usually indicated by the end of data stream symbol at the physical layer. A loopback interface is a virtual interface that is always active and reachable as long as at least one of the switch's IP interfaces is up and running. This is important due to it's address persistance, whereas interfaces or addresses of a device may change, the loopback's address doesn't.

3.2 Experiment 2

3.2.1 Objectives

In this experiment, 2 virtual LANs will be setup on the switch. VLAN30 composed by previously configurated tux33 and tux34, and VLAN31 composed by tux32. These virtual LANs will stop the tux32 from accessing tux33 and tux34, and vice-versa, even though they are all connected to the same switch.

3.2.2 Network architecture

This experiment will use the architecture described in the experiment 1 with the addition of tux32 whose eth0 interface will also be connected to the switch.

3.2.3 Main configuration commands

The configuration of tux33 and tux34 is the same as described in the previous experiment, and tux32 will be configurated in a similar fashion:

```
ifconfig eth0 up                 # Activates eth0
ifconfig eth0 172.16.31.1/24     # Configures it's ip address on eth0
```

The switch also needs to be configurated which can be done by connecting one of the tux's serial ports to the switch controller. After accessing the Switch's terminal, to create the VLANs use:

```
configure terminal  # Configure terminal
vlan 30             # Create VLAN 30
vlan 31             # Create VLAN 31
end                 # Exit config mode
```

Now the ports for each VLAN have to be specified, for simplification purposes the ports used for each tux are:

- tux32 – PORT 12
- tux33 – PORT 3
- tux34 – PORT 4

We now need to include ports 3 and 4 in VLAN30, and port 12 in VLAN31:

```
Switch# configure terminal
Switch(config)# interface fastethernet 0/3
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 30
Switch(config-if)# end
```

Listing 1: Adding port 3 to VLAN30

```
Switch# configure terminal
Switch(config)# interface fastethernet 0/4
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 30
Switch(config-if)# end
```

Listing 2: Adding port 4 to VLAN30

```
Switch# configure terminal
Switch(config)# interface fastethernet 0/12
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 31
Switch(config-if)# end
```

Listing 3: Adding port 12 to VLAN31

3.2.4 Analysis of the logs

After pinging in broadcast from tux33 and tux32 and watching where the ICMP packets are received we concluded that there are 2 broadcast domains, one per VLAN.

3.3 Experiment 3

3.3.1 Objectives

The objective of this experiment is to analyse the configuration of a cisco router, testing DNS entries and configuring routes on the local machine.

3.3.2 Network architecture

Since this was a remote experiment, there is no network architecture.

3.3.3 Main configuration commands and analysis of the logs

Setting static routing on is the fastest way to achieve communication in routed IP networks, by pointing to the next hop router which is on the way to reach destination.

```
Router# configure terminal
Router(config)# ip route {destination} 255.255.255.0 {next hop router
    address}
```

NAT can be configured in a commercial Cisco router according to the reference:

```
enable
configure terminal
ip nat inside source static local-ip global-ip
interface type number
ip address ip-address mask [secondary]
```

```
ip nat inside
exit
interface type number
ip address ip-address mask [secondary]
ip nat outside
end
```

NAT is a method that maps multiple local IPs to a public one before transferring the information. This allows an organization to have multiple devices using the same public IP that can be accessed with different ports.

To change the lookup table for hostnames we have to edit the /etc/hosts. For example, to add an entry to google with other name (youtubas) we had to add the line: 142.250.200.142 youtubas.

DNS packets were sent before ICMP's request/reply packets, upon the use of the ping command, to identify the target's IP (parlamento.pt in this case).

Concerning the Linux routing, there were ICMP packets being requested by the gateway that was routing 104.17.113.188 to the device to keep the connection alive. The routes setup on the device were the default gateway, which was removed and the route to 104.17.113.188 through 172.23.160.1 which was setup during the experiment.

3.4 Experiment 4

3.4.1 Objectives

First we need to add tux34 to VLAN31 and setup a route that allows tux33 to access tux32 through tux34. Tux32 and Tux33 should be able to access each other.

The Cisco Router should be configurated in such a way that it can access the Internet through the Lab Router, and added to VLAN31. This should allow computers that are on VLAN31 to also access the Internet. The final objective is to have access to the internet on tux33, since it can already reach devices on VLAN31 due to previous routing configuration all that's left is to also configure routes on the Cisco Router.

3.4.2 Network architecture

The architecture to be used in this experiment should be close to that used in experiment 2 with some additions. Tux34 will need to have it's eth1 interface connected to the switch, using for example PORT 14, so that it can be added to VLAN31. The Cisco router GE0 interface should be connected to the Lab Router and the GE1 interface to the switch, using for example PORT 19.

3.4.3 Main configuration commands

We now need to setup the tux34's IP address for interface eth1.

```
ifconfig eth1 up                # Activates eth1
ifconfig eth1 172.16.31.253/24  # Configures it's ip address on eth1
```

To enable port forwarding the following command must be used:

```
echo 1 > /proc/sys/net/ipv4/ip\_forward
```

The proper routes must also be setup so that tux32 and tux33 are in touch. On tux33:

```
route add -net 172.16.31.0/24 gw 172.16.30.254 # Allows access to
    172.16.31.X through tux34 (172.16.30.254)
```

On tux32:

```
route add -net 172.16.30.0/24 gw 172.16.31.253 # Allows access to
    172.16.30.X through tux34 (172.16.31.253)
```

Tux33 should now be able to communicate with every other network interface (172.16.30.254, 172.16.31.253, 172.16.31.1) this can be checked by using *ping*.

```
ping 172.16.30.254
ping 172.16.31.253
ping 172.16.31.1
```

To configure the Cisco router its controller must be connected to one of the tux's serial port. After accessing the console the following command is used to enter config mode:

```
configure terminal
```

The configurations found in the configuration file provided should be adjusted and then copied to the configuration terminal:

To configure the Interface GE0:

```
interface GigabitEthernet0/0 # Interface GE0
  ip address 172.16.2.59 255.255.255.0 # IP address and mask
  ip nat outside # NAT outside as this interface should reach the Lab
    Router
```

To configure the Interface GE1:

```
interface GigabitEthernet0/1 # Interface GE1
  ip address 172.16.51.254 255.255.255.0 # IP address and mask
  ip nat inside # NAT inside as this interface should reach the Cisco
    Switch
```

Setting up the routes to allow the Cisco Router to have access to 172.16.30.X/24:

```
ip route 172.16.50.0 255.255.255.0 172.16.51.253
```

Default gateway configuration:

```
ip route 0.0.0.0 0.0.0.0 172.16.2.254
```

The *nat pool ovrld* uses IP address 172.16.2.59 Adding the networks 172.16.30.0/24 and 172.16.31.0/24 to the access list:

```
access-list 1 permit 172.16.50.0 0.0.0.7
access-list 1 permit 172.16.51.0 0.0.0.7
```

With router's setup complete, the default gateways must be added to tux32, tux33 and tux34:

```
route add default gw 172.16.31.254 # on tux32
route add default gw 172.16.30.254 # on tux33
route add default gw 172.16.31.254 # on tux34
```

3.4.4 Analysis of the logs

Tux32 has a route that allows it to access the network 172.16.30.0 via the IP 172.16.31.253, which belongs to Tux34, and a default gw 172.16.31.254 to access the internet.

Tux33 has a route that allows it to access the network 172.16.31.0 via the IP 172.16.30.254, which belongs to Tux34, and a default gw 172.16.30.254 (tux34).

Tux34 has a default gw 172.16.31.254 to access the internet.
Forwarding table entries contain:

- The target's IP address/network
- Netmask
- Gateway
- Network Interface
- Metric

It was verified that ARP requests were sent by tux34 to determine the MAC address of tux33's eth0, tux33 also sent a request to get the MAC address of tux34.

The MAC addresses associated to the ICMP packets are the ones belonging to the origin and target devices, however due to forwarding the IP addresses associated might not be from the origin/target but from the device doing the forwarding. For example if it's between tux33 and tux32, the MAC addresses correspond to those two but the IP to tux34.

4 Conclusions

This assignment allowed us to implement an FTP client (download application) and configure a computer network. The download application was successfully tested over the configured network. The analysis and discussion of the configured network allowed us to deepen our knowledge in computer networks.

5 References

- FTP RFC – https://www.rfc-editor.org/info/rfc959
- URL RFC – https://www.rfc-editor.org/info/rfc1738
- Configuring Static Routing
- Configure NAT in a commercial router

6 Annexes

6.1 Code of the download application

```c
#include <stdio.h>

#include "parser.h"
#include "ftp.h"

int main(int argc, char *argv[]) {
    if (argc != 2) {
        printf("Usage:\n%s ftp://[<user>:<password>@]<host>/<url-path>\n", argv[0]);
        return -1;
    }

    parsed_params_t* parsed_params = parse_input_params(argv[1]);
    if (parsed_params == NULL) {
        printf("Invalid Input\n");
        printf("Usage:\n%s ftp://[<user>:<password>@]<host>/<url-path>\n", argv[0]);
        return -1;
    }

    printf("USER: \"%s\"\n", parsed_params->user);
    printf("PASSWORD: \"%s\"\n", parsed_params->password);
    printf("HOST: \"%s\"\n", parsed_params->host);
    printf("URL-PATH: \"%s\"\n\n", parsed_params->url_path);

    int socket_fd = -1;
    if ((socket_fd = ftp_setup(parsed_params->host)) < 0) {
        delete_parsed_params(parsed_params);
        return -1;
    }

    char user_anonymous[] = "anonymous";
    char pass_anonymous[] = "pass";
    char *user            = parsed_params->user;
    char *password        = parsed_params->password;

    // if no user argument
    if (user[0] == '\0') {
        user = user_anonymous;
        password = pass_anonymous;
    }

    if (ftp_login(socket_fd, user, password) != 0) {
        ftp_close(socket_fd);
        delete_parsed_params(parsed_params);
        return -1;
    }

    if (ftp_download_file(socket_fd, parsed_params->host, parsed_params->url_path) != 0) {
        ftp_close(socket_fd);
        delete_parsed_params(parsed_params);
        return -1;
    }
```

```
52
53      ftp_close(socket_fd);
54      delete_parsed_params(parsed_params);
55      return 0;
56  }
```

Listing 4: download.c

```
1  #define FTP_COMMAND_PORT 21
2
3  int ftp_setup(char *host);
4
5  int ftp_login(int socket_fd, char *user, char *pass);
6
7  int ftp_download_file(int socket_fd, char *host, char *path);
8
9  int ftp_close(int socket_fd);
```

Listing 5: ftp.h

```
1  #include "ftp.h"
2  #include "ftp_return_codes.h"
3
4  #define _GNU_SOURCE
5
6  #include <stdio.h>
7  #include <netdb.h>
8  #include <netinet/in.h>
9  #include <arpa/inet.h>
10 #include <sys/socket.h>
11 #include <string.h>
12 #include <strings.h>
13 #include <unistd.h>
14 #include <stdlib.h>
15 #include <stdbool.h>
16 #include <math.h>
17
18 #define RECV_BUFFER_START_SIZE 1000
19
20
21 static char *get_ip(char *host_name) {
22     if (host_name == NULL) {
23         return NULL;
24     }
25
26     struct hostent *h = NULL;
27
28     if ((h = gethostbyname(host_name)) == NULL) {
29         herror("gethostbyname()");
30         return NULL;
31     }
32
33     return inet_ntoa(*((struct in_addr *) h->h_addr));
34 }
35
36 static int connect_to_host(char *host_ip, uint16_t port) {
37     int sockfd;
38
39     struct sockaddr_in server_addr;
```

```
40      bzero((char *) &server_addr, sizeof(server_addr));
41      server_addr.sin_family = AF_INET;
42      server_addr.sin_addr.s_addr = inet_addr(host_ip);
43      server_addr.sin_port = htons(port);
44
45      if ((sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
46          perror("socket()");
47          return -1;
48      }
49
50      if (connect(sockfd,
51                  (struct sockaddr *) &server_addr,
52                  sizeof(server_addr)) < 0) {
53          perror("connect()");
54          return -1;
55      }
56
57      return sockfd;
58  }
59
60  // *recv_buffer needs to be free() outside after successfull return
61  static int ftp_read_line(int socket_fd, char **recv_buffer) {
62      if (recv_buffer == NULL) {
63          return -1;
64      }
65
66      size_t buffer_size = RECV_BUFFER_START_SIZE * sizeof(char);
67      char *buffer = malloc(buffer_size);
68      if (buffer == NULL) {
69          return -1;
70      }
71
72      int index = 0;
73      while (true) {
74          int res = recv(socket_fd, &buffer[index], 1, 0);
75
76          if (res == -1) {
77              return -1;
78          } else if (res == 0) {
79              return -1;
80          }
81
82          if (index > 0 && buffer[index-1] == '\r' && buffer[index] == '
    \n') {
83              buffer[index+1] = '\0';
84              break;
85          }
86
87          index += 1;
88          if (index+1 >= buffer_size) { // index+1 so that a '\0' can be
     added after
89              buffer_size += RECV_BUFFER_START_SIZE;
90              char *new_buffer = realloc(buffer, buffer_size);
91              if (new_buffer == NULL) {
92                  free(buffer);
93                  return -1;
94              }
95
96              buffer = new_buffer;
```

```
97          }
98      }
99
100     *recv_buffer = buffer;
101     return 0;
102 }
103
104 static int get_num_length(int num) {
105     return ceil(log10((double)num));
106 }
107
108 static int get_port(char *line_received) {
109     int port_msb = -1;
110     int port_lsb = -1;
111
112     if (sscanf(line_received, "227 Entering Passive Mode (%*d,%*d,%*d
    ,%*d,%d,%d)\r\n", &port_msb, &port_lsb) < 2) {
113         return -1;
114     }
115
116     return port_msb * 256 + port_lsb;
117 }
118
119 static int ftp_read_response(int socket_fd, int *port) {
120     if (socket_fd < 0) {
121         return -1;
122     }
123
124     int response_code = -1;
125     bool last_line_received = false;
126     while (!last_line_received) {
127         char *line_received = NULL;
128
129         if (ftp_read_line(socket_fd, &line_received) != 0) {
130             return -1;
131         }
132
133         printf("RECV: %s", line_received);
134
135         response_code = atoi(line_received);
136         int resp_num_digits = get_num_length(response_code);
137         if (line_received[resp_num_digits] == ' ') {
138             last_line_received = true;
139
140             if (port  != NULL) {
141                 // if wanting to retrieve port from pasv return
142                 *port = get_port(line_received);
143             }
144         }
145
146         free(line_received);
147         line_received = NULL;
148     }
149
150     return response_code;
151 }
152
153 int ftp_setup(char *host_name) {
154     if (host_name == NULL) {
```

```
155         return -1;
156     }
157 
158     char *host_ip = get_ip(host_name);
159     if (host_ip == NULL) {
160         return -1;
161     }
162 
163     int socket_fd_command = connect_to_host(host_ip, FTP_COMMAND_PORT)
    ;
164     if (socket_fd_command < 0) {
165         return -1;
166     }
167 
168     if (ftp_read_response(socket_fd_command, NULL) !=
    FTP_CODE_SERVICE_READY_FOR_NEW_USER) {
169         return -1;
170     }
171 
172     return socket_fd_command;
173 }
174 
175 static int ftp_send_command(int socket_fd, char *command, char *arg) {
176     if (command == NULL || arg == NULL) {
177         return -1;
178     }
179 
180     int cmd_size = snprintf(NULL, 0, "%s %s\r\n", command, arg);
181     if (cmd_size == -1) {
182         return -1;
183     }
184 
185     char *cmd = malloc(cmd_size + 1); // +1 for '\0'
186     if (cmd == NULL) {
187         return -1;
188     }
189 
190     if (snprintf(cmd, cmd_size + 1, "%s %s\r\n", command, arg) < 0 ) {
191         free(cmd);
192         return -1;
193     }
194 
195     if (send(socket_fd, cmd, cmd_size, 0) != cmd_size) {
196         free(cmd);
197         return -1;
198     }
199     printf("SENT: %s", cmd);
200 
201     free(cmd);
202     return 0;
203 }
204 
205 int ftp_login(int socket_fd, char *user, char *pass) {
206     if (user == NULL || pass == NULL) {
207         return -1;
208     }
209 
210     if (ftp_send_command(socket_fd, "USER", user) != 0) {
211         return -1;
```

```
212     }
213
214     if (ftp_read_response(socket_fd, NULL) !=
        FTP_CODE_USER_NAME_OKAY_NEED_PASSWORD) {
215         return -1;
216     }
217
218     if (ftp_send_command(socket_fd, "PASS", pass) != 0) {
219         return -1;
220     }
221
222     if (ftp_read_response(socket_fd, NULL) !=
        FTP_CODE_LOGIN_SUCCESSFUL) {
223         return -1;
224     }
225
226     return 0;
227 }
228
229 static int ftp_send_passv_and_get_port(int socket_fd) {
230     int port = -1;
231
232     if (ftp_send_command(socket_fd, "PASV", "") != 0) {
233         return -1;
234     }
235
236     if (ftp_read_response(socket_fd, &port) !=
        FTP_CODE_ENTERING_PASSIVE_MODE) {
237         return -1;
238     }
239
240     return port;
241 }
242
243 static int ftp_get_file(int socket_data_fd, char *path) {
244     if (path == NULL) {
245         return -1;
246     }
247
248     FILE *fp = fopen(basename(path), "w");
249     if (fp == NULL) {
250         perror("");
251         return -1;
252     }
253     int res;
254     char buffer[1000];
255     while ((res = read(socket_data_fd, buffer, 1000)) > 0) {
256         fwrite(buffer, sizeof(char), res, fp);
257     }
258
259     fclose(fp);
260     return 0;
261 }
262
263 int ftp_download_file(int socket_fd, char *host, char *path) {
264     int port = ftp_send_passv_and_get_port(socket_fd);
265
266     if (port < 0) {
267         return -1;
```

```
268     }
269
270     char *host_ip = get_ip(host);
271     if (host_ip == NULL) {
272         return -1;
273     }
274
275     int socket_data_fd = connect_to_host(host_ip, port);
276     if (socket_data_fd < 0) {
277         return -1;
278     }
279
280     if (ftp_send_command(socket_fd, "RETR", path) != 0) {
281         close(socket_data_fd);
282         return -1;
283     }
284
285     if (ftp_read_response(socket_fd, NULL) < 0) {
286         close(socket_data_fd);
287         return -1;
288     }
289
290     if (ftp_get_file(socket_data_fd, path) != 0) {
291         close(socket_data_fd);
292         return -1;
293     }
294
295     if (ftp_read_response(socket_fd, NULL) < 0) {
296         close(socket_data_fd);
297         return -1;
298     }
299
300     close(socket_data_fd);
301     return 0;
302 }
303
304 int ftp_close(int socket_fd) {
305     ftp_send_command(socket_fd, "QUIT", "");
306     ftp_read_response(socket_fd, NULL);
307
308     close(socket_fd);
309     return 0;
310 }
```

Listing 6: ftp.c

```
1  typedef struct parsed_params {
2      char *user;
3      char *password;
4      char *host;
5      char *url_path;
6  } parsed_params_t;
7
8  parsed_params_t* parse_input_params(const char* input);
9
10 void delete_parsed_params(parsed_params_t *parsed_params);
```

Listing 7: parser.h

```c
#include "parser.h"

#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <regex.h>
#include <string.h>

static const char *REGULAR_EXPRESSION = "ftp://((.+):(.+)@)?([^/@:]+)
    /([^\f\n\r\t\v\x20]*)";

#define RE_NUM_CAPTURES 6
#define RE_USER 2
#define RE_PASSWORD 3
#define RE_HOST 4
#define RE_URL_PATH 5

/*
Eg.: ftp://user:pass@ftp.up.pt/pub/kodi/timestamp.txt
0 -> ftp://user:pass@ftp.up.pt/pub/kodi/timestamp.txt
1 -> user:pass@
2 -> user
3 -> pass
4 -> ftp.up.pt
5 -> /pub/kodi/timestamp.txt
*/

static char* get_capture(const char* str, const regmatch_t *pmatch,
    uint8_t index) {
    if (pmatch == NULL) {
        return NULL;
    }

    regoff_t len = pmatch[index].rm_eo - pmatch[index].rm_so;

    char *captured_string = malloc((len + 1) * sizeof(char));
    if (captured_string == NULL) {
        return NULL;
    }

    strncpy(captured_string, str + pmatch[index].rm_so, len);
    captured_string[len] = '\0';


    return captured_string;
}

parsed_params_t* parse_input_params(const char* input) {
    regex_t     regex;
    regmatch_t  pmatch[RE_NUM_CAPTURES];

    if (regcomp(&regex, REGULAR_EXPRESSION, REG_EXTENDED) != 0) {
        return NULL;
    }

    if (regexec(&regex, input, RE_NUM_CAPTURES, pmatch, 0) != 0) {
        regfree(&regex);
        return NULL;
    }
```

```
58
59      regfree(&regex);
60
61      parsed_params_t * parsed_params = malloc(sizeof(parsed_params_t));
62      parsed_params->user     = get_capture(input, pmatch, RE_USER);
63      parsed_params->password = get_capture(input, pmatch, RE_PASSWORD);
64      parsed_params->host     = get_capture(input, pmatch, RE_HOST);
65      parsed_params->url_path = get_capture(input, pmatch, RE_URL_PATH);
66
67      return parsed_params;
68  }
69
70  void delete_parsed_params(parsed_params_t *parsed_params) {
71      if (parsed_params == NULL) {
72          return;
73      }
74
75      if (parsed_params->host)        free(parsed_params->user);
76      if (parsed_params->password)    free(parsed_params->password);
77      if (parsed_params->host)        free(parsed_params->host);
78      if (parsed_params->url_path)    free(parsed_params->url_path);
79
80      free(parsed_params);
81  }
```

Listing 8: parser.c

```
1  #define FTP_CODE_SERVICE_READY_FOR_NEW_USER 220
2  #define FTP_CODE_USER_NAME_OKAY_NEED_PASSWORD 331
3  #define FTP_CODE_LOGIN_SUCCESSFUL 230
4  #define FTP_CODE_SERVICE_CLOSING_CONTROL_CONNECTION 221
5  #define FTP_CODE_ENTERING_PASSIVE_MODE 227
```

Listing 9: ftp_return_codes.c

6.2 Logs captured

No.	Time	Source	Destinatic	Protocol	Length	Info
1	0	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
2	0.05574	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
3	1.006408	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
4	2.030409	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
5	2.055437	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
6	4.060497	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
7	5.005095	HewlettP_	Broadcast	ARP	42	Who has 172.16.1.1? Tell 172.16.1.32
8	6.03041	HewlettP_	Broadcast	ARP	42	Who has 172.16.1.1? Tell 172.16.1.32
9	6.065207	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
10	7.054411	HewlettP_	Broadcast	ARP	42	Who has 172.16.1.1? Tell 172.16.1.32
11	7.290389	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
12	8.070111	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
13	10.01019	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
14	10.07502	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
15	10.49474	Cisco_3a:f	CDP/VTP/	CDP	435	Device ID: tux-sw3 Port ID: FastEthernet0/7
16	11.02241	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
17	12.04641	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
18	12.08502	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
19	14.08475	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
20	15.01535	HewlettP_	Broadcast	ARP	42	Who has 172.16.1.1? Tell 172.16.1.32
21	16.04641	HewlettP_	Broadcast	ARP	42	Who has 172.16.1.1? Tell 172.16.1.32
22	16.08971	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
23	17.07041	HewlettP_	Broadcast	ARP	42	Who has 172.16.1.1? Tell 172.16.1.32
24	17.30309	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
25	18.10267	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
26	20.02048	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
27	20.11864	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
28	21.03841	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
29	22.06241	HewlettP_	Broadcast	ARP	42	Who has 193.136.28.10? Tell 172.16.1.32
30	22.1211	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
31	24.12602	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
32	25.02247	HewlettP_	Broadcast	ARP	42	Who has 172.16.1.1? Tell 172.16.1.32

12	6.443446	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=11/2816, ttl=64 (no response found!)
13	7.467441	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=12/3072, ttl=64 (no response found!)
14	8.019592	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
15	8.491439	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=13/3328, ttl=64 (no response found!)
16	9.515444	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=14/3584, ttl=64 (no response found!)
17	10.02442	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
18	10.53945	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=15/3840, ttl=64 (no response found!)
19	11.56345	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=16/4096, ttl=64 (no response found!)
20	12.02931	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
21	12.28979	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
22	12.58745	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=17/4352, ttl=64 (no response found!)
23	13.61144	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=18/4608, ttl=64 (no response found!)
24	14.0342	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
25	14.63545	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=19/4864, ttl=64 (no response found!)
26	15.65943	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=20/5120, ttl=64 (no response found!)
27	16.03913	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
28	16.68344	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=21/5376, ttl=64 (no response found!)
29	17.70745	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=22/5632, ttl=64 (no response found!)
30	18.04399	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
31	18.73144	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=23/5888, ttl=64 (no response found!)
32	19.75544	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=24/6144, ttl=64 (no response found!)
33	20.04883	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
34	20.77944	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=25/6400, ttl=64 (no response found!)
35	21.80344	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=26/6656, ttl=64 (no response found!)
36	22.05372	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
37	22.29417	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
38	22.82745	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=27/6912, ttl=64 (no response found!)
39	23.85144	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=28/7168, ttl=64 (no response found!)
40	24.05874	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
41	24.87544	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=29/7424, ttl=64 (no response found!)
42	25.2831	Cisco_3a:f	CDP/VTP/	CDP	435	Device ID: tux-sw3 Port ID: FastEthernet0/7
43	25.90344	172.16.31.	172.16.31.	ICMP	98	Echo (ping) request id=0x7135, seq=30/7680, ttl=64 (no response found!)
44	26.06356	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009

3	3.97683	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
4	4.009742	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
5	6.014622	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
6	8.019579	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
7	10.03216	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
8	12.02929	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
9	13.98433	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
10	14.0394	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
11	16.03904	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
12	18.04395	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
13	20.05389	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
14	22.05374	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
15	23.99192	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
16	24.05858	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
17	26.06854	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
18	27.96325	172.16.31.	224.0.0.25	MDNS	160	Standard query 0x0000 PTR _nfs._tcp.local, "QM" question PTR
19	28.06845	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
20	30.07327	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
21	32.08327	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
22	33.99955	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
23	34.083	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
24	36.08796	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
25	38.09783	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
26	40.09777	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
27	42.10263	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
28	43.99891	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
29	44.11254	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
30	46.11234	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
31	46.90122	Cisco_3a:f	CDP/VTP/	CDP	435	Device ID: tux-sw3 Port ID: FastEthernet0/7
32	48.11722	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
33	50.12734	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
34	52.12699	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009
35	54.00649	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
36	54.13189	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/31/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8009

No.	Time	Source	Destinatic	Protocol	Length	Info
1	0	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
2	1.580768	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=1/256, ttl=64 (no response found!)
3	1.580958	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=1/256, ttl=64
4	2.005003	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
5	2.585487	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=2/512, ttl=64 (no response found!)
6	2.585668	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=2/512, ttl=64
7	3.609484	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=3/768, ttl=64 (no response found!)
8	3.609656	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=3/768, ttl=64
9	4.010018	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
10	4.633483	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=4/1024, ttl=64 (no response found!)
11	4.633659	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=4/1024, ttl=64
12	5.657486	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=5/1280, ttl=64 (no response found!)
13	5.657658	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=5/1280, ttl=64
14	6.014657	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
15	6.565025	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
16	6.678455	HewlettP_	HewlettP_	ARP	60	Who has 172.16.30.1? Tell 172.16.30.254
17	6.678476	HewlettP_	HewlettP_	ARP	42	172.16.30.1 is at 00:21:5a:61:24:92
18	6.681471	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=6/1536, ttl=64 (no response found!)
19	6.681616	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=6/1536, ttl=64
20	7.705484	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=7/1792, ttl=64 (no response found!)
21	7.705664	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=7/1792, ttl=64
22	8.019549	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
23	10.02453	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004

No.	Time	Source	Destinatic	Protocol	Length	Info
1	0	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
2	2.004814	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
3	4.009751	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
4	5.586807	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
5	6.014568	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
6	8.019549	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
7	10.02459	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
8	12.02925	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
9	14.03443	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
10	15.5912	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
11	16.03899	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
12	18.04392	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
13	20.0488	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
14	22.05368	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
15	24.05857	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004

No.	Time	Source	Destinatic	Protocol	Length	Info
1	0	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
2	1.038649	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=1/256, ttl=64 (no response found!)
3	1.038825	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=1/256, ttl=64
4	1.32547	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
5	2.004892	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
6	2.05021	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=2/512, ttl=64 (no response found!)
7	2.050344	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=2/512, ttl=64
8	3.074199	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=3/768, ttl=64 (no response found!)
9	3.074334	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=3/768, ttl=64
10	4.009813	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
11	4.098191	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=4/1024, ttl=64 (no response found!)
12	4.09832	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=4/1024, ttl=64
13	5.122196	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=5/1280, ttl=64 (no response found!)
14	5.122334	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=5/1280, ttl=64
15	6.014675	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
16	6.101753	HewlettP_	HewlettP_	ARP	60	Who has 172.16.30.1? Tell 172.16.30.254
17	6.101774	HewlettP_	HewlettP_	ARP	42	172.16.30.1 is at 00:21:5a:61:24:92
18	6.146189	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=6/1536, ttl=64 (no response found!)
19	6.146338	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=6/1536, ttl=64
20	7.170198	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=7/1792, ttl=64 (no response found!)
21	7.170367	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=7/1792, ttl=64
22	8.019685	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
23	8.194205	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=8/2048, ttl=64 (no response found!)
24	8.194374	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=8/2048, ttl=64
25	9.218194	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=9/2304, ttl=64 (no response found!)
26	9.218376	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=9/2304, ttl=64
27	10.02444	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
28	10.24219	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=10/2560, ttl=64 (no response found!)
29	10.24236	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=10/2560, ttl=64
30	11.2662	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=11/2816, ttl=64 (no response found!)
31	11.26638	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=11/2816, ttl=64
32	11.33309	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
33	12.02966	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
34	12.29019	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=12/3072, ttl=64 (no response found!)
35	12.29032	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=12/3072, ttl=64
36	13.3142	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=13/3328, ttl=64 (no response found!)
37	13.31434	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=13/3328, ttl=64
38	14.03421	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
39	14.33819	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=14/3584, ttl=64 (no response found!)
40	14.33833	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=14/3584, ttl=64
41	15.36219	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=15/3840, ttl=64 (no response found!)
42	15.36237	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=15/3840, ttl=64
43	16.03939	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
44	16.3862	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=16/4096, ttl=64 (no response found!)
45	16.38633	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=16/4096, ttl=64
46	17.4102	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=17/4352, ttl=64 (no response found!)
47	17.41033	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=17/4352, ttl=64
48	18.04399	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
49	18.43421	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=18/4608, ttl=64 (no response found!)
50	18.43434	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=18/4608, ttl=64
51	19.4582	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=19/4864, ttl=64 (no response found!)
52	19.45834	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=19/4864, ttl=64
53	20.05398	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
54	20.4822	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=20/5120, ttl=64 (no response found!)
55	20.48237	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=20/5120, ttl=64
56	21.33259	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
57	21.5062	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=21/5376, ttl=64 (no response found!)
58	21.50633	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=21/5376, ttl=64
59	22.05375	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
60	22.53019	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=22/5632, ttl=64 (no response found!)
61	22.53033	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=22/5632, ttl=64
62	23.5542	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=23/5888, ttl=64 (no response found!)
63	23.55434	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=23/5888, ttl=64
64	24.05862	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
65	24.5782	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=24/6144, ttl=64 (no response found!)
66	24.57834	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=24/6144, ttl=64
67	25.6022	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=25/6400, ttl=64 (no response found!)
68	25.60237	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=25/6400, ttl=64
69	26.06857	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
70	26.6262	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=26/6656, ttl=64 (no response found!)
71	26.62633	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=26/6656, ttl=64
72	27.6502	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=27/6912, ttl=64 (no response found!)
73	27.65034	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=27/6912, ttl=64
74	28.06838	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
75	28.6742	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=28/7168, ttl=64 (no response found!)
76	28.67433	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=28/7168, ttl=64
77	29.6982	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=29/7424, ttl=64 (no response found!)
78	29.69834	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=29/7424, ttl=64
79	30.07333	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
80	30.7222	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=30/7680, ttl=64 (no response found!)
81	30.72237	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=30/7680, ttl=64
82	31.34008	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply

No.	Time	Source	Destinatic	Protocol	Length	Info
1	0	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
2	2.003844	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
3	2.565464	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
4	4.008818	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
5	6.013248	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
6	7.593674	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=1/256, ttl=64 (no response found!)
7	7.593718	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=1/256, ttl=64
8	8.019654	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
9	8.598354	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=2/512, ttl=64 (no response found!)
10	8.598393	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=2/512, ttl=64
11	9.622311	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=3/768, ttl=64 (no response found!)
12	9.622345	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=3/768, ttl=64
13	10.02389	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
14	10.64627	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=4/1024, ttl=64 (no response found!)
15	10.64631	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=4/1024, ttl=64
16	11.67024	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=5/1280, ttl=64 (no response found!)
17	11.67027	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=5/1280, ttl=64
18	12.02823	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
19	12.57778	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
20	12.69102	HewlettP_	HewlettP_	ARP	42	Who has 172.16.30.1? Tell 172.16.30.254
21	12.69117	HewlettP_	HewlettP_	ARP	60	172.16.30.1 is at 00:21:5a:61:24:92
22	12.69417	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=6/1536, ttl=64 (no response found!)
23	12.69419	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=6/1536, ttl=64
24	13.71816	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x7334, seq=7/1792, ttl=64 (no response found!)
25	13.7182	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x7334, seq=7/1792, ttl=64
26	14.03272	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
27	16.03902	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
28	18.0435	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
29	20.04767	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
30	22.05192	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
31	22.5716	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
32	24.05642	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005

No.	Time	Source	Destinatic	Protocol	Length	Info
1	0	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
2	2.004575	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
3	4.009958	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
4	5.566353	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
5	6.014168	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
6	8.019263	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
7	10.02388	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
8	12.02862	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
9	14.03351	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
10	15.57352	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
11	16.03822	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
12	18.04309	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
13	20.04785	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005

No.	Time	Source	Destinatic	Protocol	Length	Info
1	0	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=9/2304, ttl=64 (no response found!)
2	4.42E-05	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=9/2304, ttl=64
3	0.806295	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
4	1.023953	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=10/2560, ttl=64 (no response found!)
5	1.023989	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=10/2560, ttl=64
6	2.047929	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=11/2816, ttl=64 (no response found!)
7	2.047967	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=11/2816, ttl=64
8	2.114864	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
9	2.811437	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
10	3.071867	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=12/3072, ttl=64 (no response found!)
11	3.071876	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=12/3072, ttl=64
12	4.095843	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=13/3328, ttl=64 (no response found!)
13	4.095853	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=13/3328, ttl=64
14	4.815898	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
15	5.119794	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=14/3584, ttl=64 (no response found!)
16	5.119802	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=14/3584, ttl=64
17	6.143767	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=15/3840, ttl=64 (no response found!)
18	6.143805	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=15/3840, ttl=64
19	6.820997	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
20	7.167721	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=16/4096, ttl=64 (no response found!)
21	7.16773	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=16/4096, ttl=64
22	8.191682	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=17/4352, ttl=64 (no response found!)
23	8.191692	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=17/4352, ttl=64
24	8.825621	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
25	9.215653	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=18/4608, ttl=64 (no response found!)
26	9.215662	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=18/4608, ttl=64
27	10.23961	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=19/4864, ttl=64 (no response found!)
28	10.23962	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=19/4864, ttl=64
29	10.83554	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
30	11.26357	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=20/5120, ttl=64 (no response found!)
31	11.26361	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=20/5120, ttl=64
32	12.11397	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
33	12.28752	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=21/5376, ttl=64 (no response found!)
34	12.28753	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x74b4, seq=21/5376, ttl=64
35	12.83514	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8005
36	13.31148	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x74b4, seq=22/5632, ttl=64 (no response found!)

No.	Time	Source	Destinatic	Protocol	Length	Info
1	0	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
2	1.495525	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x72b7, seq=1/256, ttl=64 (reply in 3)
3	1.4957	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x72b7, seq=1/256, ttl=64 (request in 2)
4	2.004782	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
5	2.511659	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x72b7, seq=2/512, ttl=64 (reply in 6)
6	2.511831	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x72b7, seq=2/512, ttl=64 (request in 5)
7	2.902388	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
8	3.535651	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x72b7, seq=3/768, ttl=64 (reply in 9)
9	3.535789	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x72b7, seq=3/768, ttl=64 (request in 8)
10	4.014744	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
11	4.55965	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x72b7, seq=4/1024, ttl=64 (reply in 12)
12	4.559786	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x72b7, seq=4/1024, ttl=64 (request in 11)
13	5.583659	172.16.30.	172.16.30.	ICMP	98	Echo (ping) request id=0x72b7, seq=5/1280, ttl=64 (reply in 14)
14	5.583793	172.16.30.	172.16.30.	ICMP	98	Echo (ping) reply id=0x72b7, seq=5/1280, ttl=64 (request in 13)
15	6.014617	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
16	6.575619	HewlettP_	HewlettP_	ARP	42	Who has 172.16.30.254? Tell 172.16.30.1
17	6.575737	HewlettP_	HewlettP_	ARP	60	172.16.30.254 is at 00:21:5a:5a:7d:74
18	6.72593	HewlettP_	HewlettP_	ARP	60	Who has 172.16.30.1? Tell 172.16.30.254
19	6.725946	HewlettP_	HewlettP_	ARP	42	172.16.30.1 is at 00:21:5a:61:24:92
20	8.019391	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
21	10.02939	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
22	12.0292	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
23	12.90673	Cisco_3a:f	Cisco_3a:f	LOOP	60	Reply
24	14.03923	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
25	16.039	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
26	18.04385	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
27	20.05389	Cisco_3a:f	Spanning-	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004

18	11.00074506	172.16.30.1	172.16.31.253	ICMP	98	Echo (ping) request id=0x794d, seq=2/512, ttl=64 (reply in 19)
19	11.00088293	172.16.31.253	172.16.30.1	ICMP	98	Echo (ping) reply id=0x794d, seq=2/512, ttl=64 (request in 18)
20	12.02474801	172.16.30.1	172.16.31.253	ICMP	98	Echo (ping) request id=0x794d, seq=3/768, ttl=64 (reply in 21)
21	12.02488693	172.16.31.253	172.16.30.1	ICMP	98	Echo (ping) reply id=0x794d, seq=3/768, ttl=64 (request in 20)
22	12.02962275	Cisco_3a:fa:84	Spanning-tree-(for-bridges)	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
23	13.04874992	172.16.30.1	172.16.31.253	ICMP	98	Echo (ping) request id=0x794d, seq=4/1024, ttl=64 (reply in 24)
24	13.04888534	172.16.31.253	172.16.30.1	ICMP	98	Echo (ping) reply id=0x794d, seq=4/1024, ttl=64 (request in 23)
26	15.12092499	HewlettP_5a:7d:7	HewlettP_61:24:92	ARP	60	Who has 172.16.30.1? Tell 172.16.30.254
27	15.12094567	HewlettP_61:24:9	HewlettP_5a:7d:74	ARP	42	172.16.30.1 is at 00:21:5a:61:24:92
28	15.30543861	172.16.30.254	224.0.0.251	MDNS	160	Standard query 0x0000 PTR _nfs._tcp.local, "QM" question PTR _ftp.
29	15.70694868	Cisco_3a:fa:84	Cisco_3a:fa:84	LOOP	60	Reply
30	16.03433873	Cisco_3a:fa:84	Spanning-tree-(for-bridges)	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
31	16.25320631	172.16.30.1	172.16.31.1	ICMP	98	Echo (ping) request id=0x7951, seq=1/256, ttl=64 (reply in 32)
32	16.25347206	172.16.31.1	172.16.30.1	ICMP	98	Echo (ping) reply id=0x7951, seq=1/256, ttl=63 (request in 31)
33	17.1767095	HewlettP_61:24:9	HewlettP_5a:7d:74	ARP	42	Who has 172.16.30.254? Tell 172.16.30.1
34	17.17682467	HewlettP_5a:7d:7	HewlettP_61:24:92	ARP	60	172.16.30.254 is at 00:21:5a:5a:7d:74
35	17.27273982	172.16.30.1	172.16.31.1	ICMP	98	Echo (ping) request id=0x7951, seq=2/512, ttl=64 (reply in 36)
36	17.27299516	172.16.31.1	172.16.30.1	ICMP	98	Echo (ping) reply id=0x7951, seq=2/512, ttl=63 (request in 35)
37	18.04421726	Cisco_3a:fa:84	Spanning-tree-(for-bridges)	STP	60	Conf. Root = 32768/30/fc:fb:fb:3a:fa:80 Cost = 0 Port = 0x8004
38	18.29674382	172.16.30.1	172.16.31.1	ICMP	98	Echo (ping) request id=0x7951, seq=3/768, ttl=64 (reply in 39)
39	18.29699525	172.16.31.1	172.16.30.1	ICMP	98	Echo (ping) reply id=0x7951, seq=3/768, ttl=63 (request in 38)
40	19.32073958	172.16.30.1	172.16.31.1	ICMP	98	Echo (ping) request id=0x7951, seq=4/1024, ttl=64 (reply in 41)

205	163.7373162	HewlettP_61:24:92	Broadcast	ARP	60	Who has 172.16.30.254? Tell 172.16.30.1
206	163.7373418	HewlettP_5a:7d:74	HewlettP_61:24:92	ARP	42	172.16.30.254 is at 00:21:5a:5a:7d:74
207	163.7374771	172.16.30.1	172.16.31.1	ICMP	98	Echo (ping) request id=0x7b9a, seq=1/256, ttl=64 (reply in 208)
208	163.7377174	172.16.31.1	172.16.30.1	ICMP	98	Echo (ping) reply id=0x7b9a, seq=1/256, ttl=63 (request in 207)
209	163.7374914	Kye_25:26:0a	Broadcast	ARP	42	Who has 172.16.31.1? Tell 172.16.31.253
210	163.7376023	HewlettP_61:30:63	Kye_25:26:0a	ARP	60	172.16.31.1 is at 00:21:5a:61:30:63
211	163.7376071	172.16.30.1	172.16.31.1	ICMP	98	Echo (ping) request id=0x7b9a, seq=1/256, ttl=63 (reply in 212)
212	163.7377128	172.16.31.1	172.16.30.1	ICMP	98	Echo (ping) reply id=0x7b9a, seq=1/256, ttl=64 (request in 211)