V 1.1 8th -June-2023

Universidad de León
Ingeniería Informática
Course on Computer Networks

Homework On UDP Datagram Sockets

Details about this homework submission are published in the agora

Introduction. UDP Datagram sockets is the service interface to the UDP protocol. UDP is the Internet protocol used for the end-to-end communication among applications that don't require delivery guarantees. The protocol that provides those guarantees is TCP which we'll introduce in the Distributed Systems course which comes next September.

UDP adds a new *multiplexing layer* on top of IP, one that permits the identification of applications interested in communicating over Internet. The multiplexing keys provided by UDP are known as *UDP Ports,* 16-bit unsigned integers. For example, your personal computer or your cellular phone use reserved UDP port 123 for exchanging time-of-day information with their time server and they do this quite often over any work session. This homework aims to help you to discover UDP and program a UDP client that communicates with a provided UDP server.

Skimming the basics about UDP and Sockets. UDP is named *the simple multiplexer* in the Textbook by Peterson and Davie (P&D), and it certainly is a simple protocol. Along with the textbook by P&D, you can skim the following presentation to acquaint yourselves with the protocol:

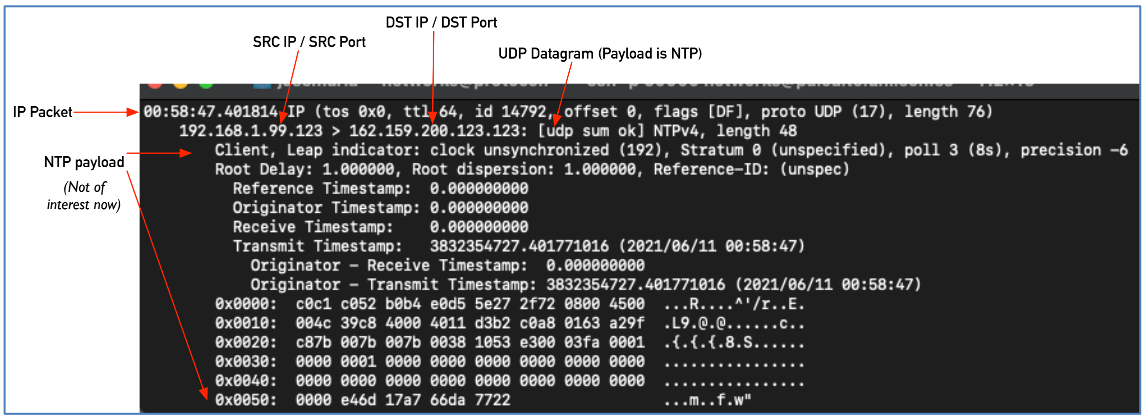
http://paloalto.unileon.es/ds/lec/BasicTCP-1.pdf

You should start at slide no. 5 and skim through slide no. 12 (Little's Law is no import at this time).

Exercise 1: NTP (Network Time Protocol) is a good example of an application protocol that uses UDP. The NTP application protocol runs on top of UDP and uses UDP port 123. When an NTP client sends a *clock synchronization request* to an NTP server, the request (An NTP request message) is encapsulated into an UDP datagram which destination port is 123. Is the source port also port 123?

We can run a tcpdump trace that captures an NTP request generated by your host and then respond to the preceding question. Execute the following command on your home computer (You should replace enp1s0 by whatever NIC name is appropriate) and let it run for a few minutes until your NTP client decides to send the NTP clock sync request. When the request is sent, you'll be able to identify the field structure of the UDP datagram that encapsulates the request, including the source and destination UDP ports used by the protocol. Observe what happened in paloalto.unileon.es in Lab B6, today:

```
$ tcpdump -XX -vvv -n udp port 123
```



```
00:58:47.401814 IP (tos 0x0, ttl 64, id 14792, offset 0, flags [DF], proto UDP (17), length 76)
    192.168.1.99.123 > 162.159.200.123.123: [udp sum ok] NTPv4, length 48
        Client, Leap indicator: clock unsynchronized (192), Stratum 0 (unspecified), poll 3 (8s), precision -6
        Root Delay: 1.000000, Root dispersion: 1.000000, Reference-ID: (unspec)
          Reference Timestamp:  0.000000000
          Originator Timestamp: 0.000000000
          Receive Timestamp:    0.000000000
          Transmit Timestamp:   3832354727.401771016 (2021/06/11 00:58:47)
            Originator - Receive Timestamp:  0.000000000
            Originator - Transmit Timestamp: 3832354727.401771016 (2021/06/11 00:58:47)
        0x0000:  c0c1 c052 b0b4 e0d5 5e27 2f72 0800 4500  ...R....^'/r..E.
        0x0010:  004c 39c8 4000 4011 d3b2 c0a8 0163 a29f  .L9.@.@......c..
        0x0020:  c87b 007b 007b 0038 1053 e300 03fa 0001  .{.{.{.8.S......
        0x0030:  0000 0001 0000 0000 0000 0000 0000 0000  ................
        0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0050:  0000 e46d 17a7 66da 7722                 ...m..f.w"
```

Figure 1. tcpdump trace of an NTP request

The NTP payload encapsulated into the UDP datagram is not of our interest at this time (It has been printed out in dark red). Observe the source and destination IP addresses and the respective ports printed out *in blue*. Now you can tell for sure that the NTP client sent an NTP request to the server using source port and destination port 123.

If your operating system is taking a long time to send the NTP clock sync request, install the ntpdate command (# apt install ntpdate) and execute it while you have the tcpdump trace running in a different window:

```
# ntpdate -s es.pool.ntp.org
```

Do this exercise at your own computer, and include the UDP datagram containing the NTP request and identify its *five* fields: Src port, dst port, checksum, length and payload. If you need more information in the trace, search for convenient options of tcpdump that might be of help.

Exercise 2. Explain what mechanism allows the Linux traceroute utility to sort the list of hops to the provided destination. Provide sufficient evidence.

Exercise 3. Calculate the smallest IP Block that encompasses prefixes 192.168.2.64/26 and 192.168.1.0/24.

Exercise 4. Can IP blocks 192.168.2.64/26 and 192.168.2.128/26 be joined? Provide a demonstration to support your solution.

Exercise 5. Can IP blocks 192.168.2.128/26 and 192.168.2.192/26 be joined? Provide a demonstration to support your solution.

Exercise 6. Write a summary and a short discussion to the following paragraph from the book "IP Address Management. Principles and Practice" © 2011 by WILEY/IEEE Press, Timothy Rooney.

14.3.5 Block/Subnet Joins

A join combines two contiguous same-sized address blocks or subnets into a single block or subnet. We saw an example of joining blocks in the block deletion section. After freeing up the Anaheim block, 10.32.142.0/24, we joined it to a contiguous free block, 10.32.143.0/24 to create a single 10.32.142.0/23 block. Successive joins may be performed to consolidate smaller chunks of contiguous address space. Joins are only valid for contiguous blocks of the same size. Joining a /25 and a /24 is not valid as the "other /25" not included in the join must remain uniquely identified. However, two contiguous /25s and a neighboring /24 can be joined to form a /23. The two /25s would be joined first to form a /24; then this and the other /24 can be joined to create a /23.

Rolling up of joined blocks may also require an updating of DNS reverse zones to consolidate underlying device resource records into a "joined" reverse zone reflecting the resulting consolidated subnet.

Exercise 7. Write a command line utility that establishes whether connectivity exists between two hosts connected to the same LAN. The utility should remind of the IP-based ping utility. Write the Client and the Server programs for the Linux operating system. Explain your choice of language and the general analysis considerations that you have made.