**EECE5155: Wireless Sensor Networks and the Internet of Things**

**Computer Laboratory Assignment 1**

**Authors:**

* Ziyu Bian
* Liangshe Li
* Computer Lab Team 21

**Date:** 10/3/2020

**Task 1**

The requirement of the issue is shown as the following:

文本, 信件, 电子邮件

描述已自动生成

It is obvious that we are going to establish the point-to-point network in the issue. First of all, we will construct the network topology we are going to achieve, which is shown as the following:

192.168.2.0/24

N0 ------------------------- N1

1. **Experimental setup:**

This section we are going to modify and explain the code.

Firstly, we will take various of head files, by the way, we will make the claim to the name space:

文本

描述已自动生成

In the next part, we are going to claim the log component whose name is “FirstScriptExample”, we can turn on or turn off the output of the log component with the method of quoting the name, the code is just shown as the following:

手机屏幕的截图

描述已自动生成

The major part of the construction of the network is the main function, we will make the illustration in sequence.

There are two lines of the code to make the two log components valid, these two components are established within the application of Echo Client and Echo Server. Furthermore, we will set the log component with the level of LOG\_LEVEL\_INFO. This part is shown as the following:

文本

描述已自动生成

After that, we will focus ourselves on the network topology implement. First, we will create the object of ns3 node, they are on behalf of the computer in the simulating process. We establish the “Nodecontainer” with the name of nodes. As the image of topology shown before, there are two nodes in the whole network, that means the value of the parameter of the Create method is 2. The code is shown as the following:

图形用户界面, 文本

描述已自动生成

After the nodes are established, we are going to create the connection between the points, we will use “PointToPointHelper” to help us to deploy and connect the PointToPointNetDevice and the object of PointToPointChannel. According to the issue, the transmission rate is 10 Mbps while the delay is 2ms. The code is just shown as the following:

文本

描述已自动生成

After the nodes are connected, we will install the protocol stack for the PointToPointHelper object and the point-to-point net device. Finally, we will arrange the IP address for the device of the node. The last part we will use the Ipv4AddressHelper to combine the IP addresses with the corresponding device. The code of the part is shown as the following:

文本

描述已自动生成

The next part we are going to work out is the application program. We will set the application of UdpEchoServer on our nodes, the parameter’s value is the port number. Then we use the method of Start and Stop to set the time interval. According to the issue, the port number is 63, so the code we modified is shown as the following:

图形用户界面, 文本

描述已自动生成

After the set of the UdpEchoServer, we begin to deal with the UdpEchoClient. We will use UdpEchoClientHelper to help us to set the application of client.

We firstly create the object of client, telling it that set its remote address as the server node’s IP address. Meanwhile we make it send the package to the port 63. For the part of attribute setting, three attributes we will set, they are the maximized number of packages we are going to deliver, the time interval between two packages and the size of data contained by each package. We also use the Start and Stop instruction to tell the client when to begin and when to stop. The code we modified is shown as the following:

文本

描述已自动生成

Finally, we run the simulator and destroy all the objects created to free the space. To enable to trace the package, we will use the promiscuous mode to trace the data. The code is shown as the following:

图片包含 文本

描述已自动生成

1. **Results:**

After running the first.cc files consisting of codes above, we will get two .pcap files:

First-0-0.pcap

图片包含 图形用户界面

描述已自动生成

First-1-0.pcap:

图形用户界面, 文本, 应用程序, 电子邮件

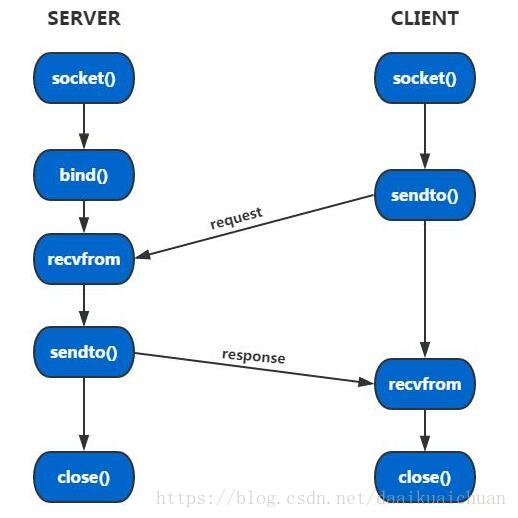
描述已自动生成

First of all, we need to make the illustration to the name of the pcap file. The name can be understood from the following formula:

Example-<Node Index>-<Net Device Index>.pcap. From the two files above, we can clearly see that the tracing work for two nodes are executed by the same net device. Then, if we look at the communication between the two nodes, we can find several interesting things. First, we can see the communicating process， first of all, the size of package delivered is not equal to the size of the data, since Udp transmission needs to capsulate the data. First reason is that it will add the IP header which is 22 bytes. Second reason is it will add a Udp header whose size is 8 bytes. Furthermore, since the point-to-point protocol also requires one header, that makes the package delivered has a bigger size.

Furthermore, we can see the mode of the UDP transmission, let us look at the image shown above.

It is obvious that we set the node 0 as the client and the node 1 as the server, and the time interval between two messages node 0 received is larger than the time interval node 1 has. Since the server requires some time to deal with the requirement, and the frame of the Udp transmission is shown as the following:



Let us make the deeper analysis to the file:

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

图形用户界面, 文本, 电子邮件

描述已自动生成

We can find that checksum is validation disabled, the reason is that the receiver needs to re-compute the checksum and make the comparation to see if they are equal, so here the checksum should be ignored.

1. **Learnt Lessons:**

In this first task, we have simulated a point-to-point link using C++ and have known how to use Wireshark to show the condition of the network by opening ,pcap files.

**Task 2**

The real image of the network we are going to achieve is shown as the following:

图示, 日程表

描述已自动生成

The network contains:

1. 3 nodes in the first shared but operating under CSMA
2. 3 nodes in the second shared but operating under CSMA
3. 2 nodes in the point to point link, and furthermore, node2 and node3 have two network interfaces, one for each link they are connected.

The application running in the network are:

1. UDP Echo Server at Node 1: Listening on port 21.
2. UDP Echo Client at Node 5: Sends 2 UDP Echo packets to the server at times 4s and 7s.

We are required to use wireshark to capture and analyse the packet traces.

To implement the goal of the laboratory assignment, we will first design the network is shown as the following:

192.168.3.0

N0 N1 N2 ---------------N3 N4 N5

| | | | | |

= = = = = = = = = = = = = =

LAN 192.168.1.0. LAN 192.168.2.0

As the topology image shown above, what we need to do at the issue is build the CSMA network as well as point-to-point network, we will use second.cc file as the basement and just make some adjustment to make it adopted to the issue, next part we will explain the code in sequence just in order to make it clear. Finally, we will use show the packet trace and make the analysis.

However, we actually get the two CSMA networks connecting with each other like the structure below:

192.168.3.0

N3 N2 N0 ---------------N1 N4 N5

| | | | | |

= = = = = = = = = = = = = =

LAN 192.168.1.0. LAN 192.168.2.0

The cause why the serial numbers of nodes changes is that we first generate two p2p nodes and then we generate two CSMA networks on each of p2p nodes. This will be presented as the codes below.

**1. Experimental setup:**

As usual, we will firstly begin the code with various head files as well as defining the log component, just shown as the following:

文本

描述已自动生成

After that, in the main function, we will use verbose flag to judge whether the log components of UdpEchoClientApplication and UdpEchoSeeverApplication are enabled to use, if it is true, then both of them can be used, so we will set the verbose flag as True. Afterwards, since the property of CSMA network is that there must be two more nodes in each cluster, the value of nCsma should not less than 1. In the issue, we can clearly see that there are 3 nodes in each CSMA cluster, and that means the value of nCsma is 3 minus 1 which equals to 2. Then we can create the object of the class CommandLine, then add the attribute of nCsma and verbose to its parameter system which is shown as the following:

文本

描述已自动生成

To ensure that the two values of nCsma we set is bigger than zero, we use the two line of the code just in case:

图形用户界面

描述已自动生成

The next part we are about to create the object of NodeContainer p2pNodes and csmaNodes. As we have already known, point-to-point network in the issue contains the node 2 and node 3, that means, the parameter in the instruction of p2pNode.Create() is 2. Then we create the CSMA network clusters. Since there are two clusters according to the issue, and the index of client and server in each cluster is 1 and 2, the code we corrected is shown as the following (the cause of changes of serial numbers):

文本

描述已自动生成

Afterwards, we establish the channel for each network and add the corresponding attribute of data transmission rate and time delay towards the nodes just shown as the following:

文本

描述已自动生成

Next part we are going to deal with the protocol stack, we will combine the network of point-to-point with the network of CSMA, then we will arrange the IP address for the networks just like the issue’s requirement.

The code for the part is just shown as the following:

文本

描述已自动生成

After setting up the network layer, we begin to establish the transmission layer, this problem asks us to use the protocol of UDP, according to the issue’s requirement, the UDPEchoServer do the listening work on port 21, so we will arrange the port 21 to the server, and we will also add the corresponding attribute to the client and server:

文本

描述已自动生成

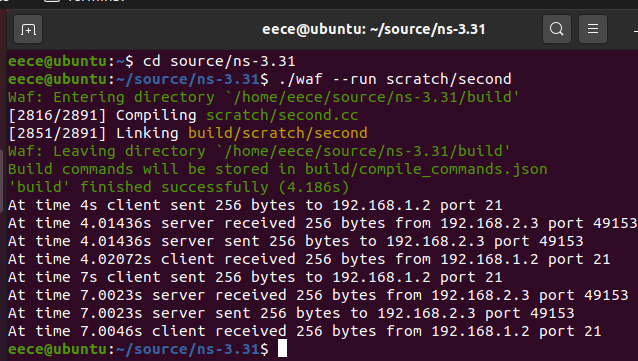
Finally, we will create the routing table for each node of the network, we will create pcap tracing file to trace point-to-point network, by the way, we will trace CSMA network in promiscuous mode. We will run the simulator for the last step.

文本

描述已自动生成

1. **Results:**

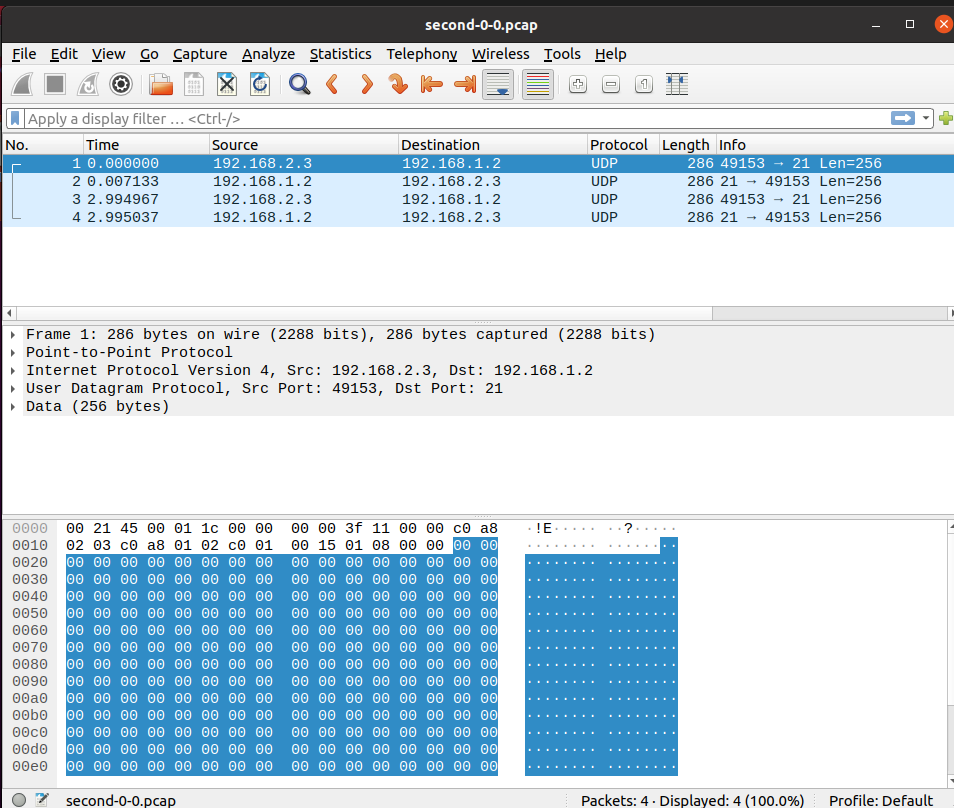
After running the second,cc file consisting of codes above, we will get the screen beow and three .pcap files.



As you see, at time 4s and 7s the client sends 256 bytes to the server(192.168.1.2, it is the second node in the Csma1 network). And the port of the server is 21. The port of the client is 49153, the same with task 1.

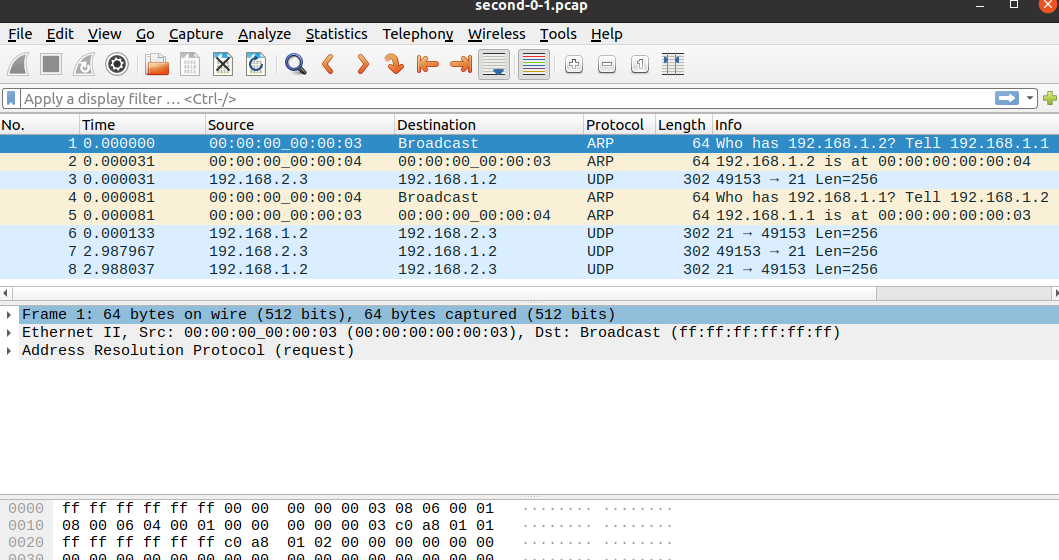
And then we get second-0-0.pcap, second-0-1.pcap, second-4-0.pcap. First we will explain why we get this three .pcap files. As the task 2 requests, we need to trace Node2 and Node4. And as we refer above, the Node2 in the homework picture is actually Node0 in our codes (Because we first set up two p2p nodes), and Node4 in the homework picture is the same Node4 in our codes. In the following we will use the Node number in our codes (not homework) to illustrate. The Node0 has two interfaces, one is p2pDevices and the other is csmaDevices1. So the p2pDevices of Node0 generate second-0-0.pcap, the csmaDevices1 of Node0 generate second-0-1.pcap (It is the second net device). And the csmaDevices2 of Node4 generate second-4-0.pcap(It has only one net device).

Second-0-0.pcap:



It is a p2p node and we have illustrated it in the task1, so we will not explain more.

Second-0-1.pcap:



This part we can see there is the new protocol, ARP (Address Resolution Protocol). It is a TCP/IP protocol used for getting the physical address. The host broadcast the ARP request containing IP address of the destination to all the devices of the network and receive the returned message as well just so that it can ensure the goal address. It will also store the IP address and the physical address to the ARP buffer for a while just in order to save the time and resource.

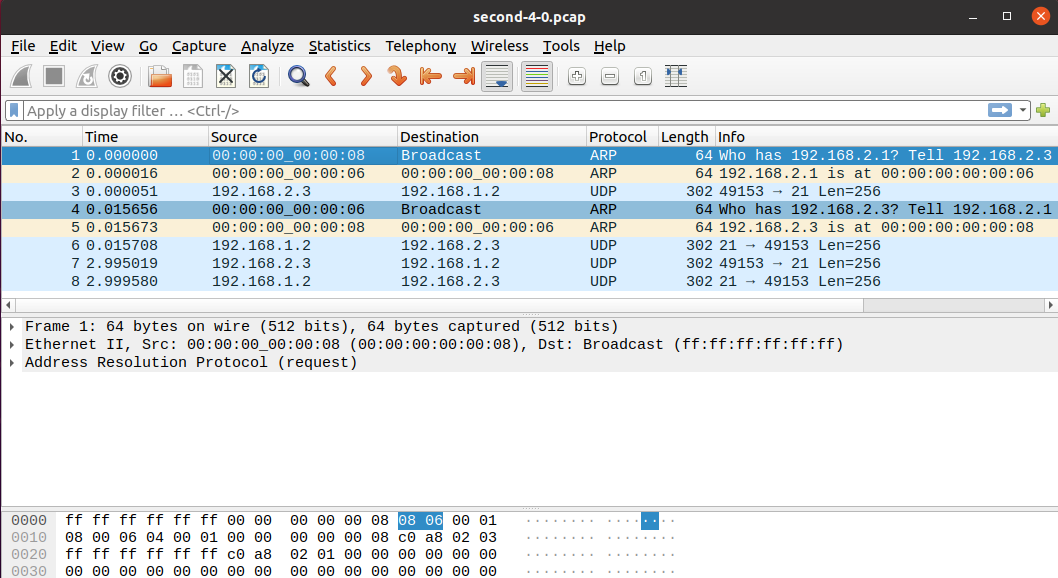
For instance, at time 1 0.00000 the host whose MAC address is 00:00:00\_00:00:03 will send to an ARP message to ask all the devices (as you see in the datagram “ff ff ff ff ff ff”, in TCP/IP it means broadcast) whose IP address (server’s address) is 192.168.1.2. And then the device whose MAC address is 00:00:00\_00:00:04 answers the call and it sends a returning message to 00:00:00\_00:00:03 at time 2 0.000031. And then the host will send the UDP datagram to the destination at time 3 0.000081.

We can also see that at the second time when the client sends 256 bytes to the server, the host (00:00:00\_00:00:03) do not send ARP message to the destination. This is because the host has already known the goal address, so there is no need to send ARP message.

The UDP datagram is almost the same with the p2p node in second-0-0.pcap. The only difference is that it adds 16 bytes which present the Ethernet, and the Ethernet provides the MAC addresses of the source and the destination.

The second-4-0.pcap is the result of Node4. The node has only one net device, the CsmaDevices2. As we see below, it has almost the same structure with the second-0-1.pcap. The difference is that the MAC addresses of source and destination is 00:00:00\_00:00:06 and 00:00:00\_00:00:08. This is because the Node4 is in the CsmaDevices2, and Node0 is in the CsmaDevices1.

Second-4-0.pcap:



1. **Learnt Lessons:**

In this second task, we have simulated a point-to-point link whose both two nodes are CSMA networks. Same as task1, we use Wireshark to trace these packets and show the results.

As you see, we create a zip file which contains this report. In the zip file, there are also the .cc files and the .pcap files requested in the homework.