CSG 1105 / 5130 - Applied Communications

Week 4 Tutorial

Objectives

- To familiarise yourself with the Wireshark application
- Be able to identify the different layers of the OSI model in actual network traffic
- Be able to identify broadcast packets
- Understand collision detection and collision domains and it's affect on network traffic

By the end of this workshop you should be able to

- Use Wireshark to sniff network traffic and identify elements of a packet
- Tell the difference between unicast and broadcast traffic
- Be able to explain collision detection and it's affect on network traffic

Required Tools & Documents

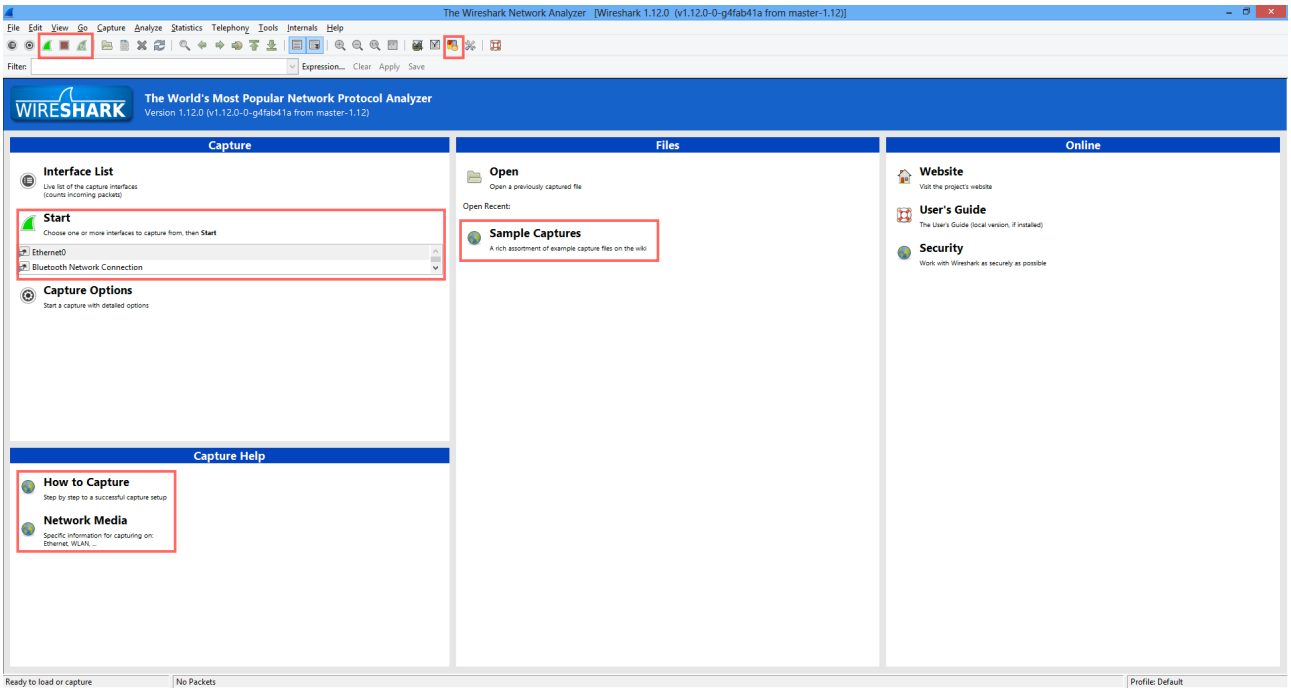
- Packet Tracer (Available in Unit Documents on Blackboard)
- Wireshark (Available in Unit Documents on Blackboard)
  - Note: Mac users also need XQuartz (included when downloaded from Blackboard)
- An internet connection
- A terminal/command prompt application with Telnet capabilities - more details on the last page of this tutorial

Optional Downloads

- A recording of this Tutorial from blackboard; discussions are held in class and clarifications are made too.

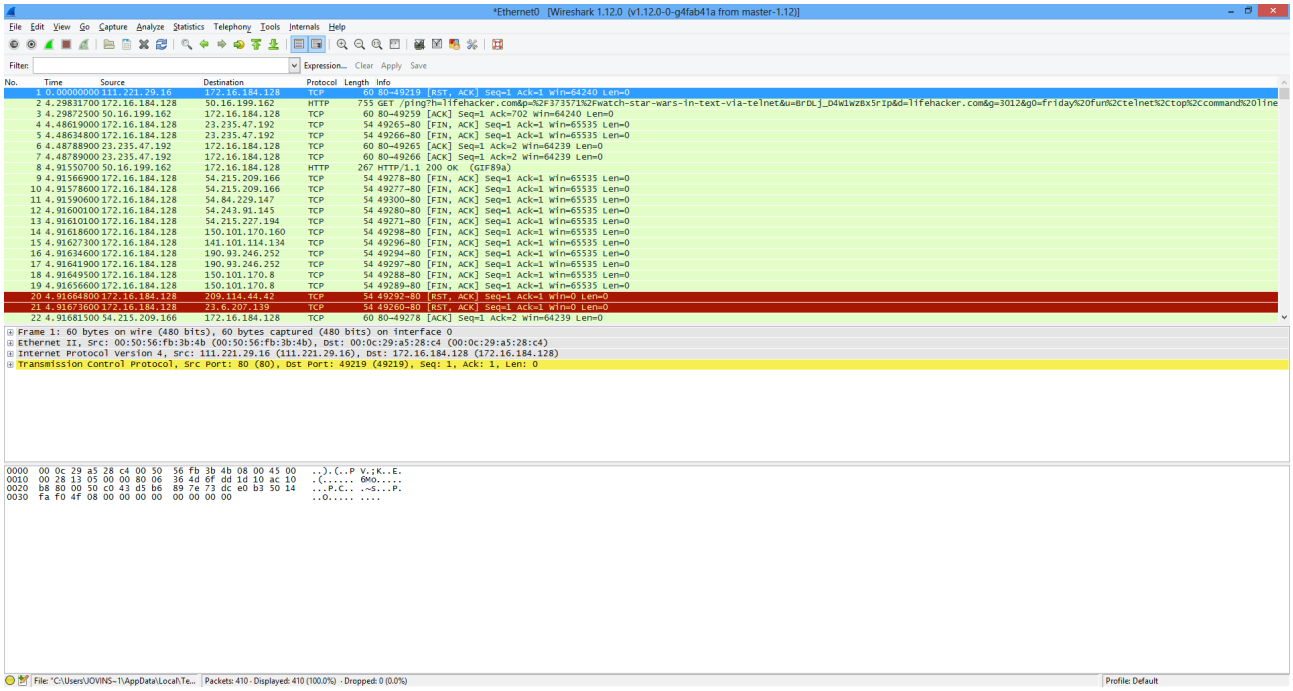
Task 1 - Wireshark Analyses

1. Open Wireshark and take a look at the user interface. On the next page is information about these key elements.



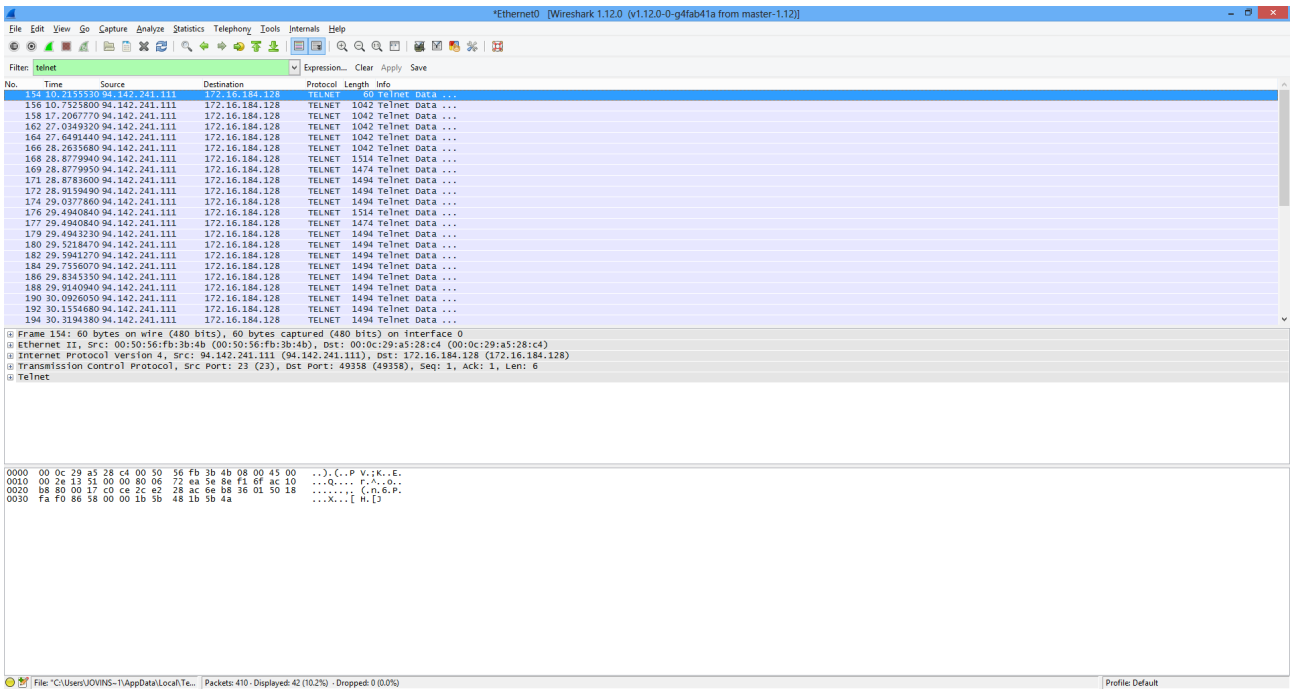
2. The top 3 icons in the left are quick access buttons to start, stop or restart a live capture of network traffic. The ‘Start’ with the interfaces below it is a way to start listening on a specific interface, or all interfaces. The ‘Sample Captures’ are examples of live traffic packets, found on the Wireshark Wiki (a very useful resource), and below is some help sites by Wireshark to aid you in listening to the traffic.
3. For now, we’ll look into capturing some live traffic. For my example I will be using ‘telnet’ which a protocol that is well known in the IT professional world to be unsecured and as such we can see the plain-text being sent across the internet. The telnet I will be using is a free online source that allows you to view Star Wars in ASCII animation. The command to connect is:

```
telnet towel.blinkenlights.nl
```



4. After a short while, stop the capture (red stop button) and the telnet connection by closing your terminal/command prompt application (you can watch through this in your spare time if you choose to). You’ll notice there is three main sections to the captured data.
    1. The list of all captured packets; this contains information such as source and destination addresses, protocol, length and info.
    2. A layered view of the selected packet we are viewing, with more information provided to us
    3. The view of the frames in hexadecimal and ASCII
5. We can order our list of captured packets by any of the columns, so if we wanted to view the largest traffic, we can sort by Length, or how many ARP or TCP requests are being made we could sort by Protocol, for now though, we’ll leave it sorted by the packet number.

6. Above our list of captured packets you'll notice a 'Filter:' field. In this field we want to track the information we just generated by our telnet request. So, let's filter only TELNET protocol packets by typing in *telnet* and then clicking on 'Apply' to the right. Below is an example of our new view:



Note: You will notice it will show a red background on the filter search until you have typed in the full word of telnet. This is because it is syntax checking your input until it finds a match.

7. If we start selecting these packets we can see the information in our layer viewer has the same structure for every telnet packet but the information contained within changes. Without clicking on the '+' next to each one we can select the layer and find the part of the frame in the frame viewer below that it responds to. If we select the first one starting with '`Frame`' you'll see the entire frame below is highlighted, moving down, when we select '`Ethernet II`' we can see what part of the frame is the Ethernet information (MAC addresses etc.). Selecting '`Internet Protocol Version 4`' shows the source and destination addresses part of the frame; selecting '`Transmission Control Protocol`' shows the where in the frame the ports and sequence information is stored. Finally, selecting '`Telnet`' shows us the section of the frame specifying that it is telnet and the data being transferred. Clicking on the '+' next to each layer allows us to see the individual information provided in each layer, clicking on that newly revealed information shows where in the frame it is stored.
8. I'll break down my example frame on the next page…

Frame Information:

0000	00 0c 29 a5 28 c4 00 50	56 fb 3b 4b 08 00 45 00	..).(..P V.;K..E.
0010	00 2e 13 51 00 00 80 06	72 ea 5e 8e f1 6f ac 10	...Q.... r.^..o..
0020	b8 80 00 17 c0 ce 2c e2	28 ac 6e b8 36 01 50 18	......,. (.n.6.P.
0030	fa f0 86 58 00 00 1b 5b	48 1b 5b 4a	...x...[H.[J

`Frame 154: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0`

Clicking the + on this part only shows information about the frame which Wireshark has created, it is not sent along with the frame over the network.

`Ethernet II, Src: 00:50:56:fb:3b:4b (00:50:56:fb:3b:4b), Dst: 00:0c:29:a5:28:c4 (00:0c:29:a5:28:c4)`

Selecting this part highlights the majority of the 0000 sequence of the frame, and clicking on the + reveals more modular information about this. Clicking on each row shows us how the information is presented in the frame. The first 6 sets of hexadecimal information (`00 0c 29 a5 28 c4`) in the long strings is our Destination MAC address. The next 6 sets of hexadecimal information (`00 50 56 fb 3b 4b`) is our Source MAC address, and then the `08 00` is identifying it as IP.

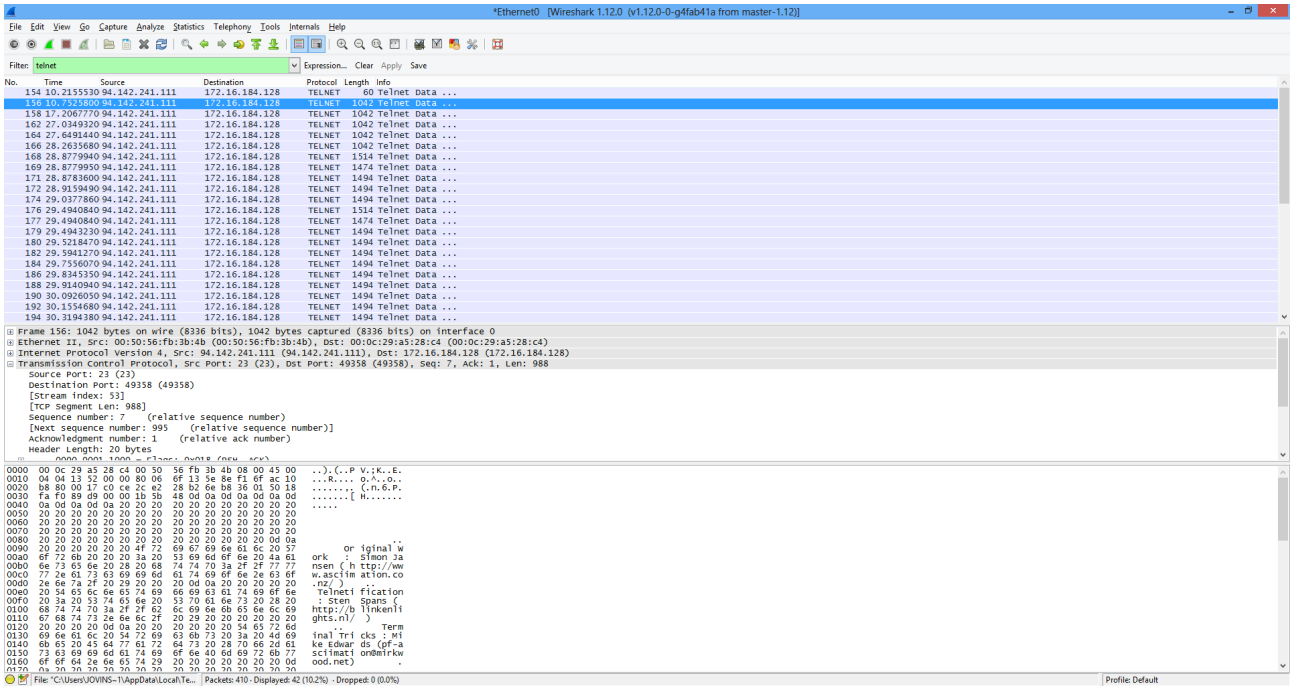
`Internet Protocol Version 4, Src: 94.142.241.111 (94.142.241.111), Dst: 172.16.184.128 (172.16.184.128)`

Diving into this part we can view information about the IPv4 aspect of the frame. clicking on each part reveals in the frame what part is represented by what hexadecimal value or ASCII value in the lower view. Here we can identify what IP version we're using, the length of the frame, the number of 'bounces' this frame can receive before it is dropped (time to live), and also a checksum, which can be used to validate the data if enabled.

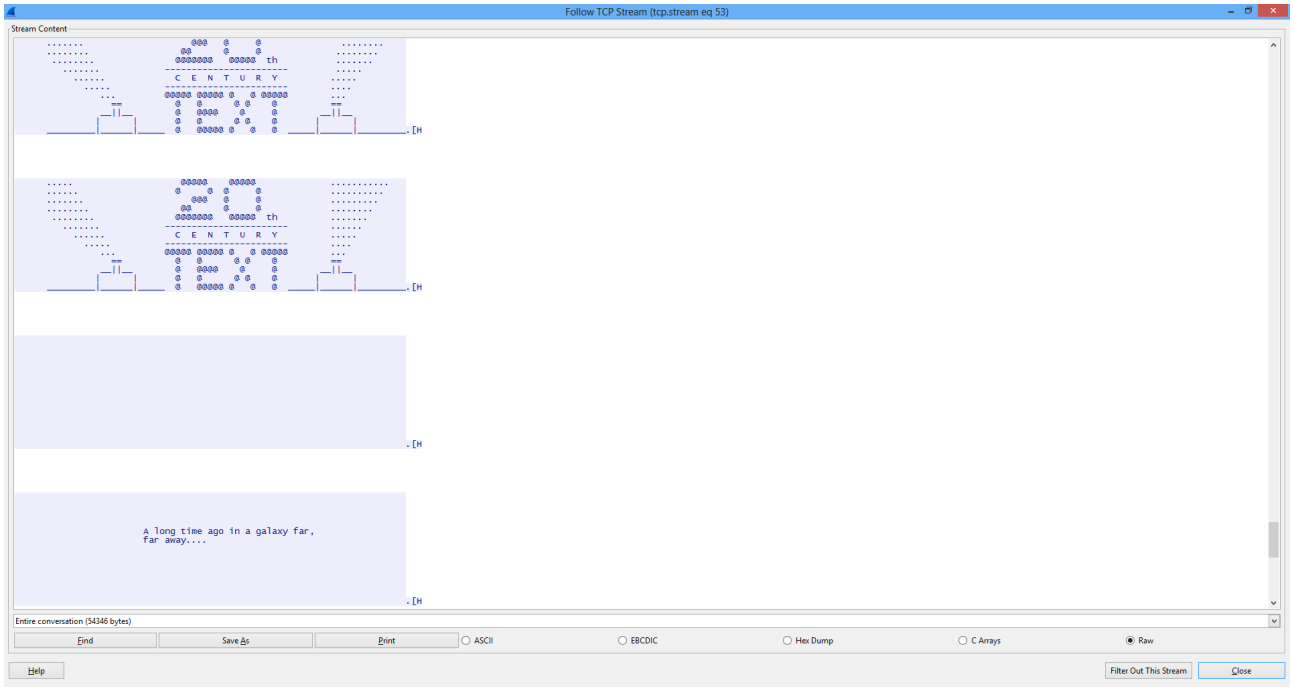
`Transmission Control Protocol, Src Port: 23 (23), Dst Port: 49358 (49358), Seq: 1, Ack: 1, Len: 6`

More specific information can be revealed on in the TCP section of our frame. Such as ports in use and the sequence of the frames. We can see that this frame is Sequence number 1 and it's Length is 6, as such we know that the next frame should show Sequence number 7 (1 + 6 = 7). This is a way for the communicating interfaces to identify when a frame is dropped or lost.

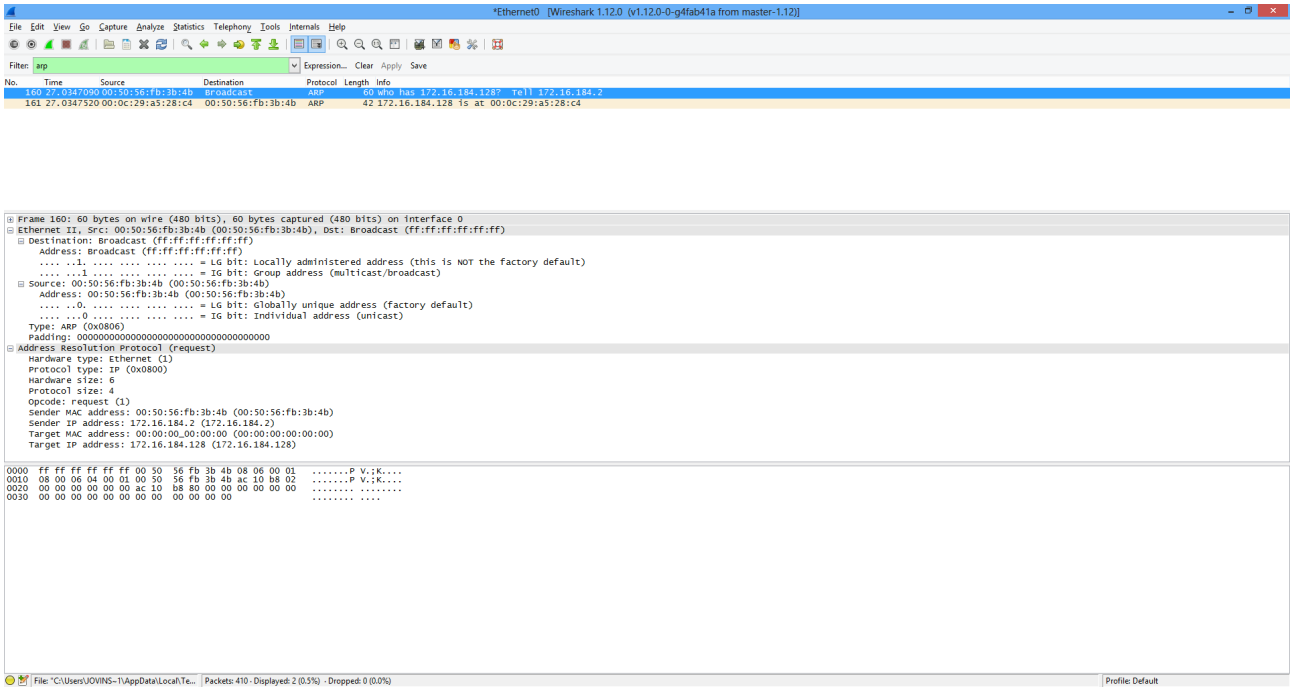
9. Now, if we continue to view each new packet that we have captured we can begin to see our telnet information form in the bottom most view, such as below, we can see ‘Original Work : Simon’ etc.



10. Here is where encryption and secure connections become important, or potentially, mission critical. If we are using telnet to connect to sensitive devices, anyone that is listening in can see all of the data that is being sent between the server and client; however, drilling through packet by packet can become tedious and tiresome. As such, we have an easier way to do this.
11. If you click on ‘Analyze’ at the top, and then go down the menu to ‘Follow TCP Stream’ you can then see each TCP packet in it’s translated final form, and effectively watch a telnet session take place. For us, this is just going to let us watch Star Wars again, but for passwords and login credentials, this would allow you to effectively steal their user identity.



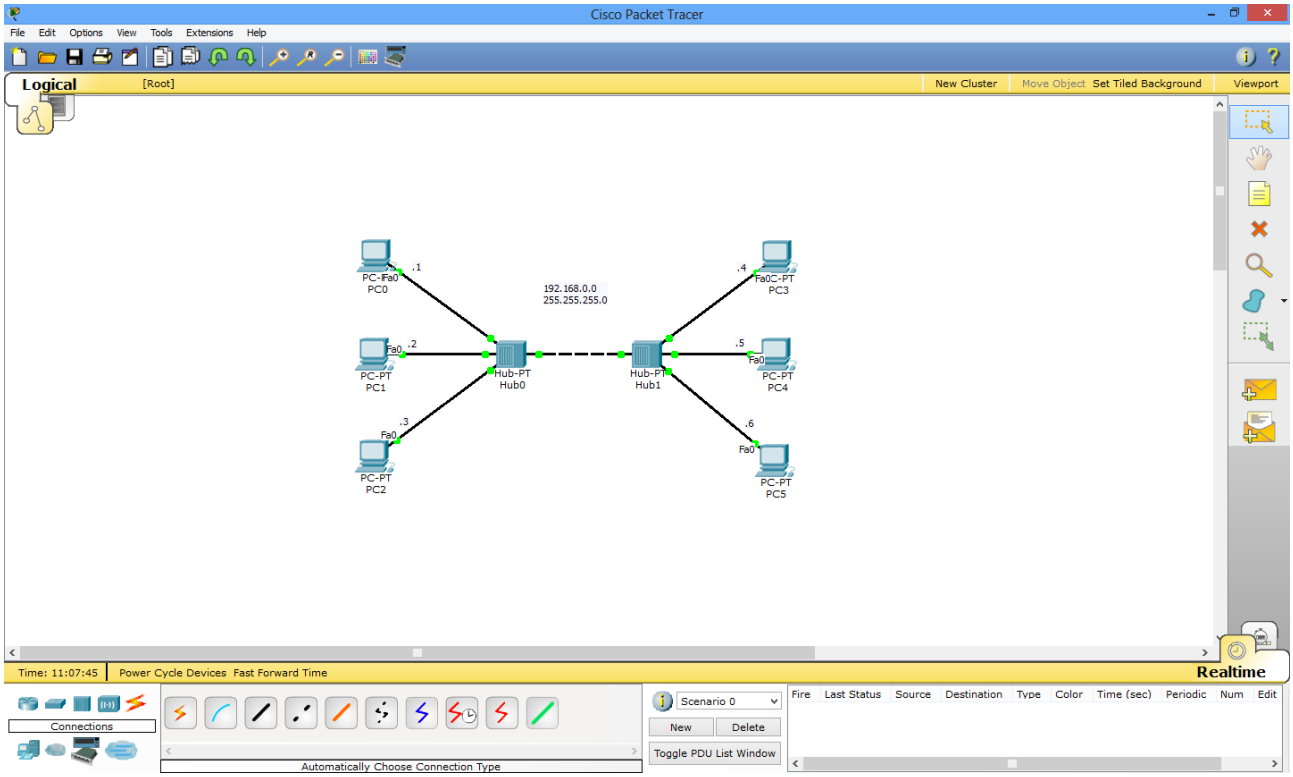
12. That was an example of unicast traffic. A source host has connected to a destination host and begun communication between the two hosts only. Now, let's analyse some broadcast traffic. You should have captured some when we were capturing before. Try filtering to find 'arp' traffic.
13. ARP is Address Resolution Protocol and it is used by networking devices and servers to build up a routing table so it can pair MAC Addresses with IP Addresses. As each MAC Address is unique to each networking interface it will lease out a temporary IP Address to it. If this routing device is not the DHCP server, it will ask who an IP Address belongs to and store it for later use. This allows devices to route traffic that is on Layer 2 or Layer 3 to the same location.
14. If we look at our ARP information we can see something along the lines of this:



15. You'll notice that with ARP communication there is two packets relating to each other. One is addressed as 'Broadcast' and the other is a unicast packet. This is how ARP requests are made. The requesting device will send this communication out to every connected device, when the mentioned IP Address is found, that device will respond to the source.
16. Drilling into the information, if we click on '`Ethernet II`' and expand it, we can see that the Broadcast is sent out at Layer 2, using only MAC Addresses, and that the broadcast address is `ff:ff:ff:ff:ff:ff`. If we were to send out a Layer 3 broadcast, we'd be making use if `255.255.255.255` which is a specially reserved address purely for this communication type.
17. If we look at the 'Address Resolution Protocol (request)' section of our frame by expanding it, we can see something curious. There is a Target IP Address existing, however, the MAC Address is simply `00:00:00:00:00:00`. This represents an incomplete routing table, and that is why the ARP request is sent out.
18. There are other examples of broadcast traffic around, such as DHCP where a newly connecting computer will broadcast a request for an IP from the server, the DHCP server will then send back to the requesting host it's assigned IP Address and request acknowledgement of this.

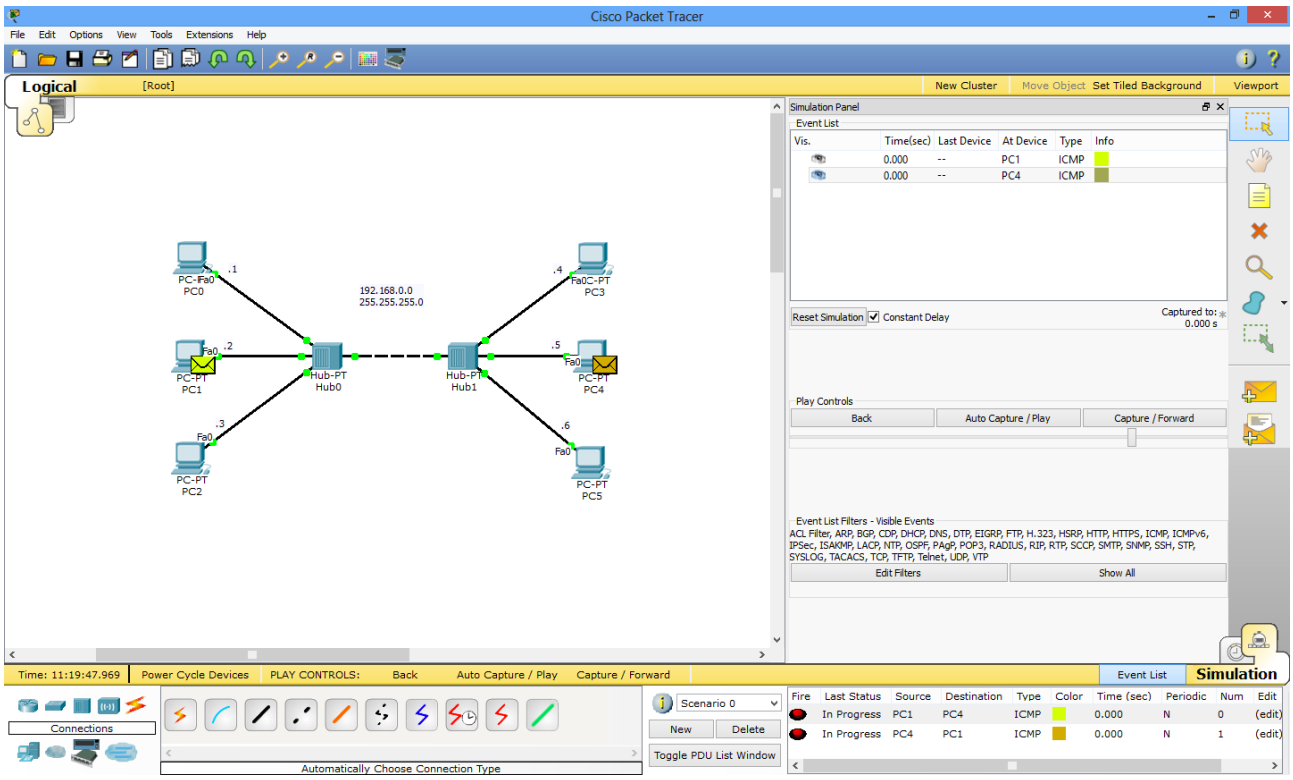
Task 2 - Packet Tracer and Collision Domains

1. Our Packet Tracer exercises this week will focus on the benefits of Switches over Hubs and displaying how the collision domains improve the functionality of a network. As such we will be looking at different configurations of networks; Firstly, two hubs, then one hub and one switch, and finally, multiple switches.
2. Create a new workspace in Packet Tracer and build the same network as Week 1's tutorial. Use the same configuration for the IP Addresses (192.168.0.0, 255.255.255.0) but don't run any of the PDU tests like at the end of Week 1. You should have the below structured then:

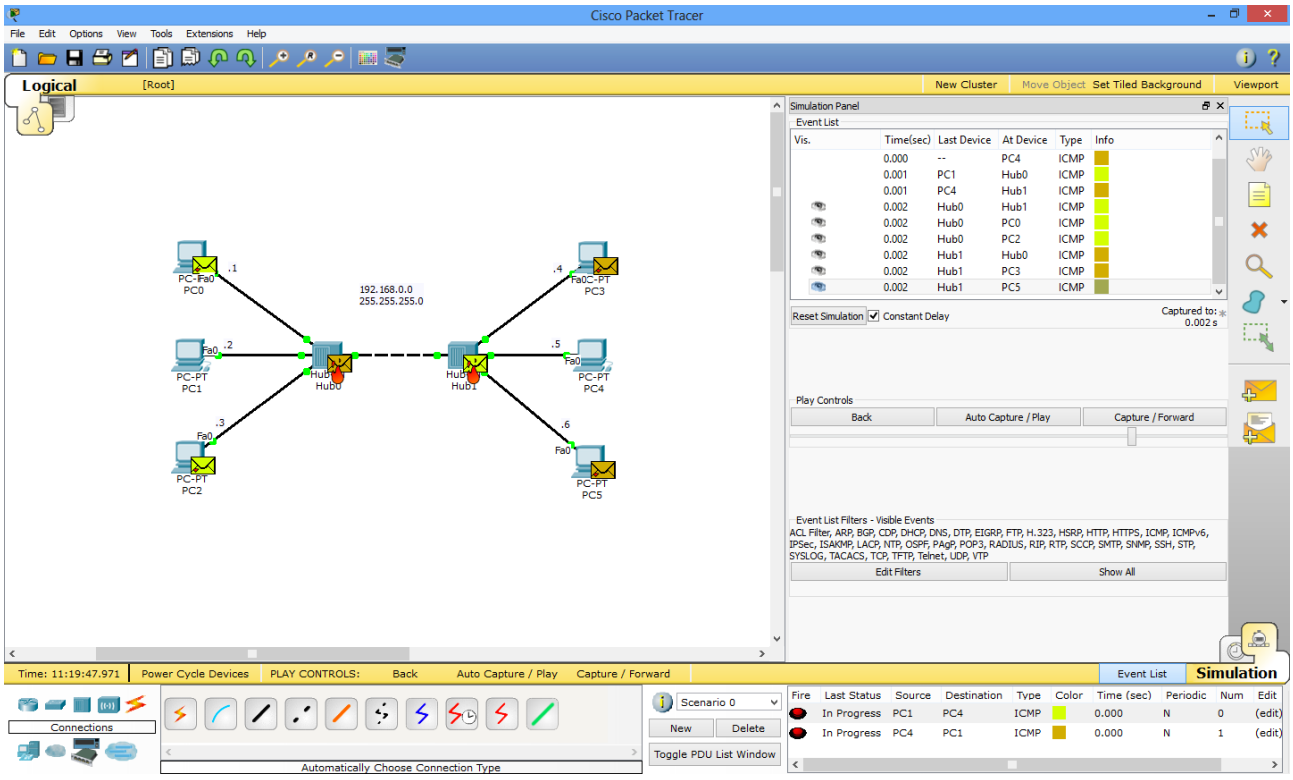


3. As previously discussed, Hubs are an OSI Layer 1 device. As such they are effectively multiport repeaters, in that they repeat bit-for-bit all the information in one port, out of every other port. They can in fact detect collisions, and have a very basic response to a collision. Let's explore that now.

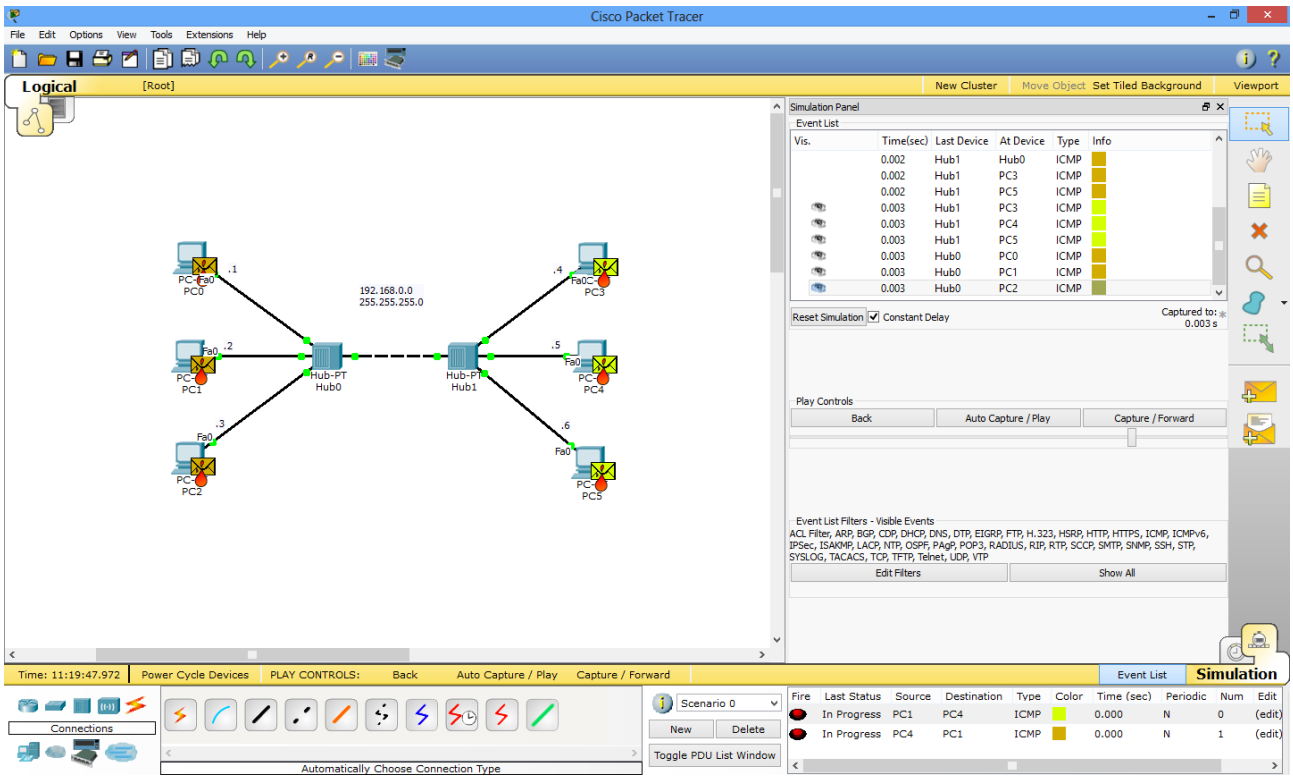
Note: The example scenarios below have been run multiple times, as such, you may see yours run an ARP request before sending the actual PDU; this is normal. You can simply delete the scenario and recreate it to see it happen without ARP occurring.



4. Create a PDU to travel from any one computer on one side of the network destined for the other, and then create a second PDU to travel in the reverse direction. Now we want to 'Capture/Forward' to view how our PDUs traverse the network. We should see a collision at some point, showing flames on the PDUs over the Hubs - like below:

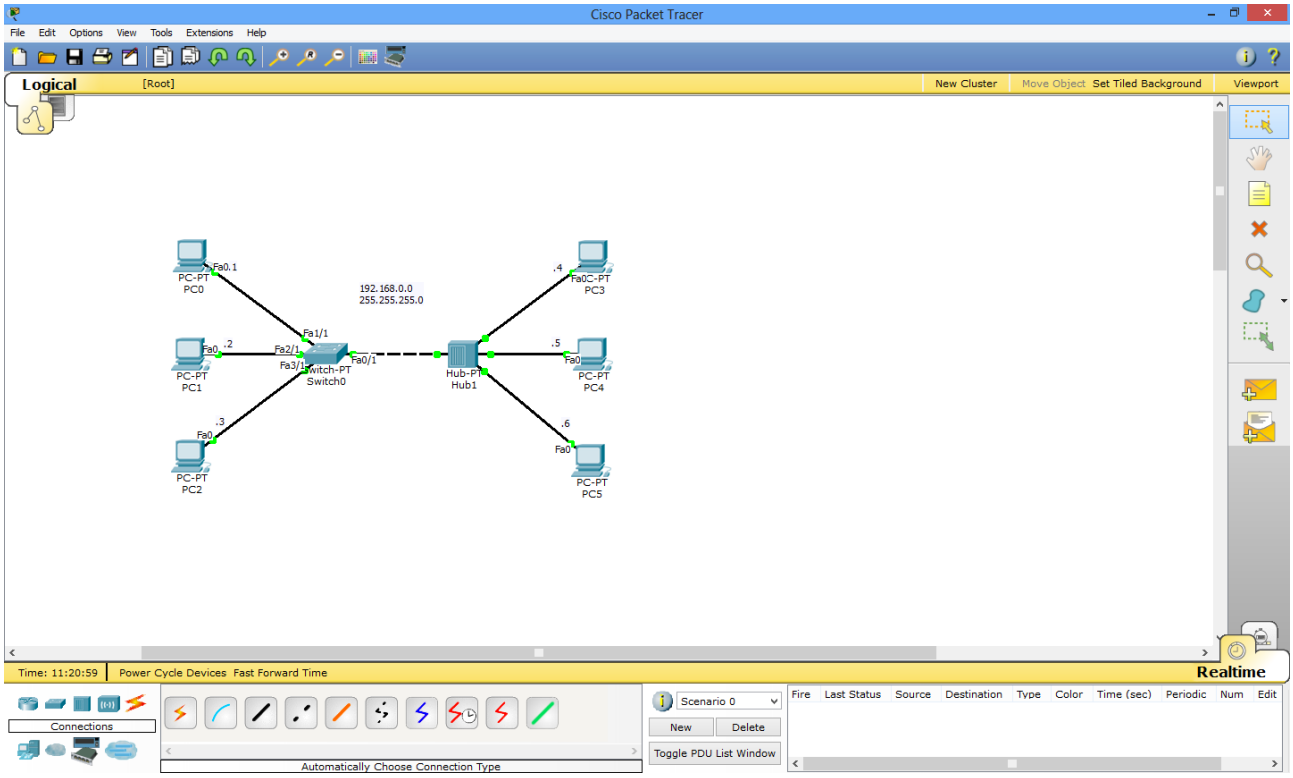


5. The flames indicate that a collision has been detected on the two hubs. Now we see what happens when you 'Capture/Foward' again. (See next page)

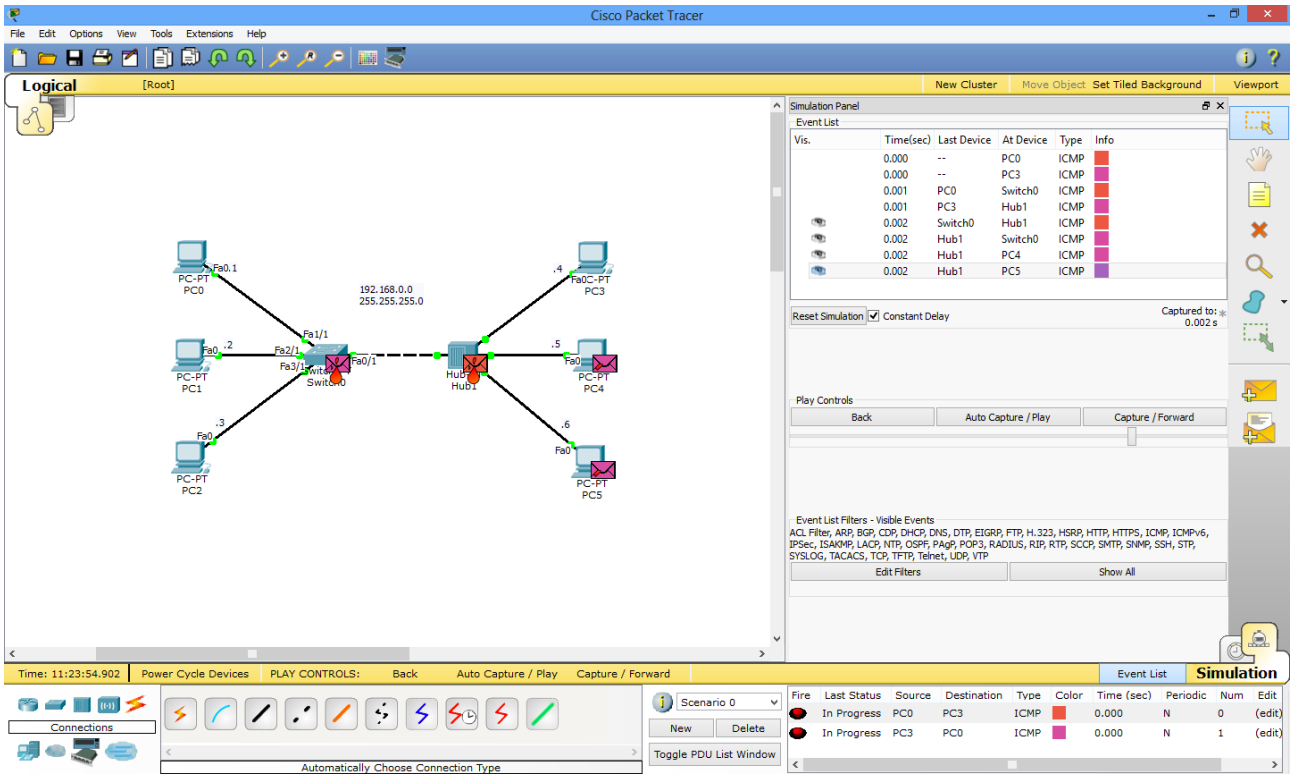


6. You'll now see a PDU has arrived at every computer with flames. This isn't Packet Tracer indicating that the PDU didn't make it due to a collision; it's showing how Hubs handle a collision. Hubs have no memory to store packets in, and as such a packet must be transmitted while it is received. This forces it to operate in half-duplex, meaning that it can only communicate in one direction at a time. This also means that they encompass a larger collision domain, and as such, collisions are far more frequent than Routers or Switches or other higher layer devices.
7. You'll notice that there is also a lot of excess network load when sending a single PDU along the network. One packet, destined for one machine gets sent to every other machine, effectively using an extra 400% of the network capacity. Now, let's investigate what happens when we replace one of our Hubs with a Switch.

8. Modify your network to now have one hub and one switch, we'll have a look at what happens to our network utilisation. Your network should now look like below:

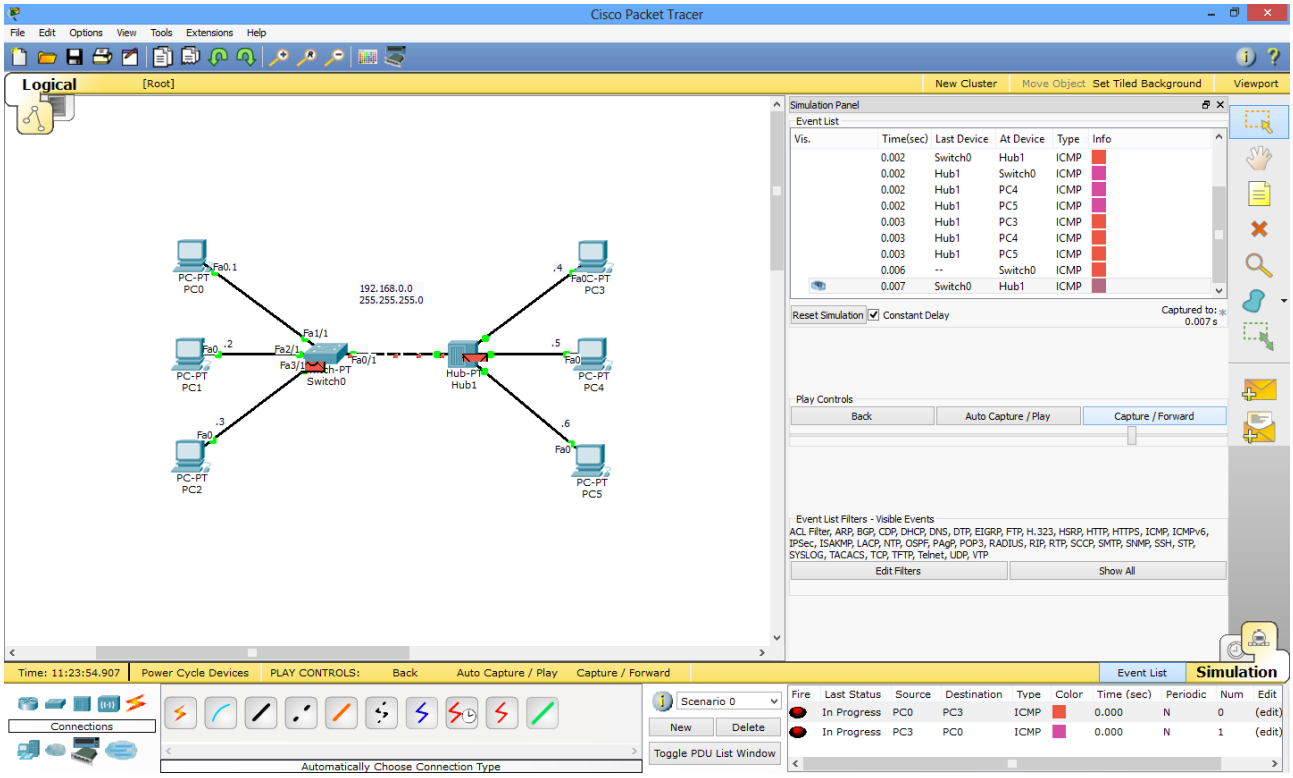


9. Now we'll do the exact same exercise as Step 4. We'll monitor what happens to the PDUs in this scenario.



10. If a Switch is meant to be superior why are we seeing this same scene where there is a collision appearing on the Switch? This is because the Switch makes use of CSMA/CD which

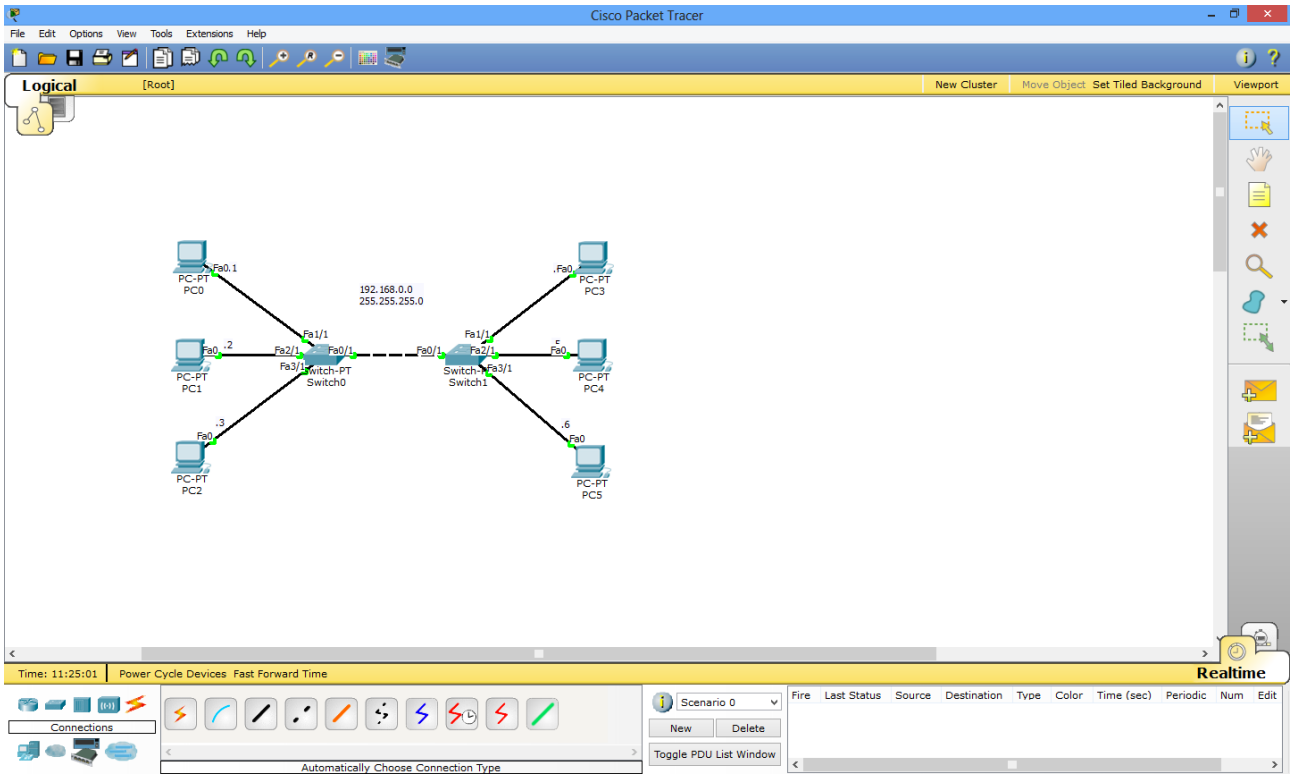
is Carrier Sense Multiple Access with Collision Detection. Essentially, the Switch understood that some traffic was entering the same time as exiting and as such prepares itself to resend its data, as we'll see if we continue. The Hub will still send out it's jamming signal as per normal however.



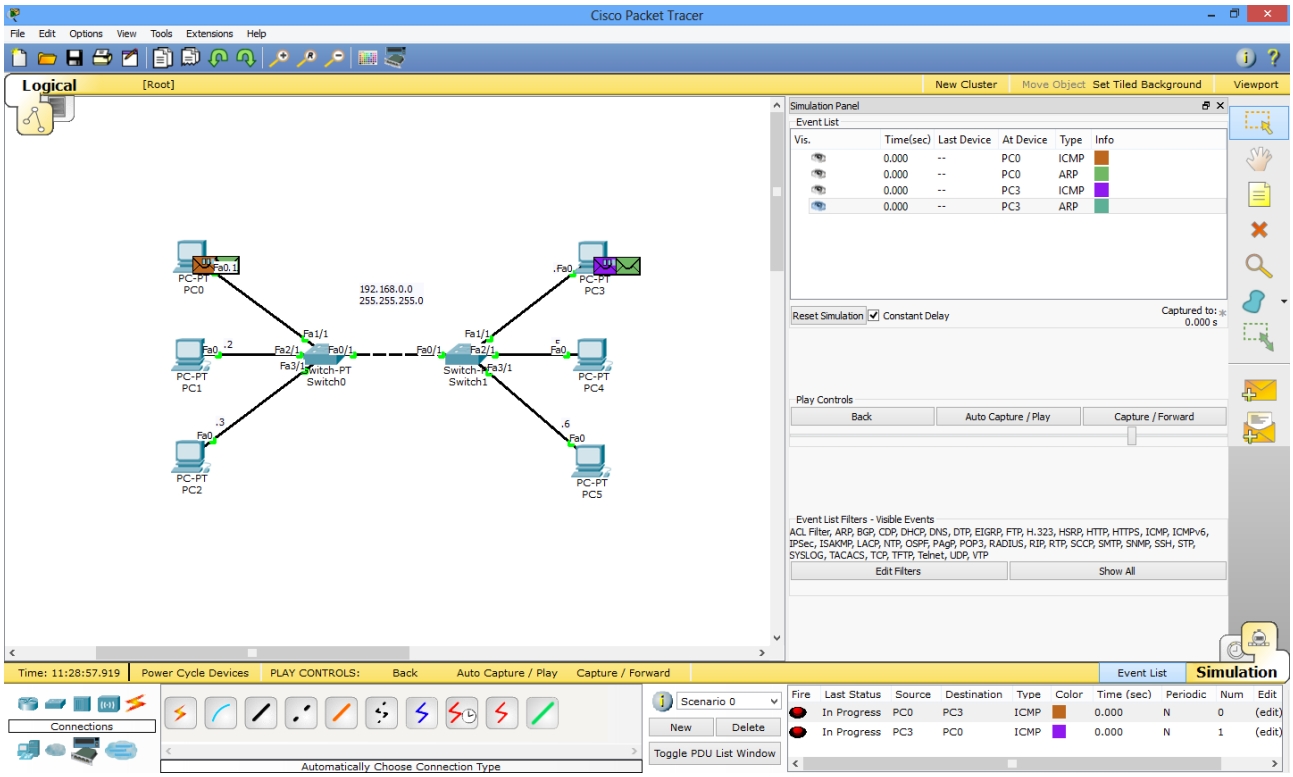
11. Now, we'll see the Hub continue it's normal behaviour of simply broadcasting everything on it's half of the network. Communications encompassing both parts of the network are now only using an additional 200% of the optimal traffic compared to the previous 400%.
12. Why doesn't the Switch do this same action?

    A switch doesn't broadcast everything because it has some memory to be able to store a Routing table where it can remember what devices are connected to what port, as such, it can simply direct all the traffic towards it's intended destination. See further on the next page.

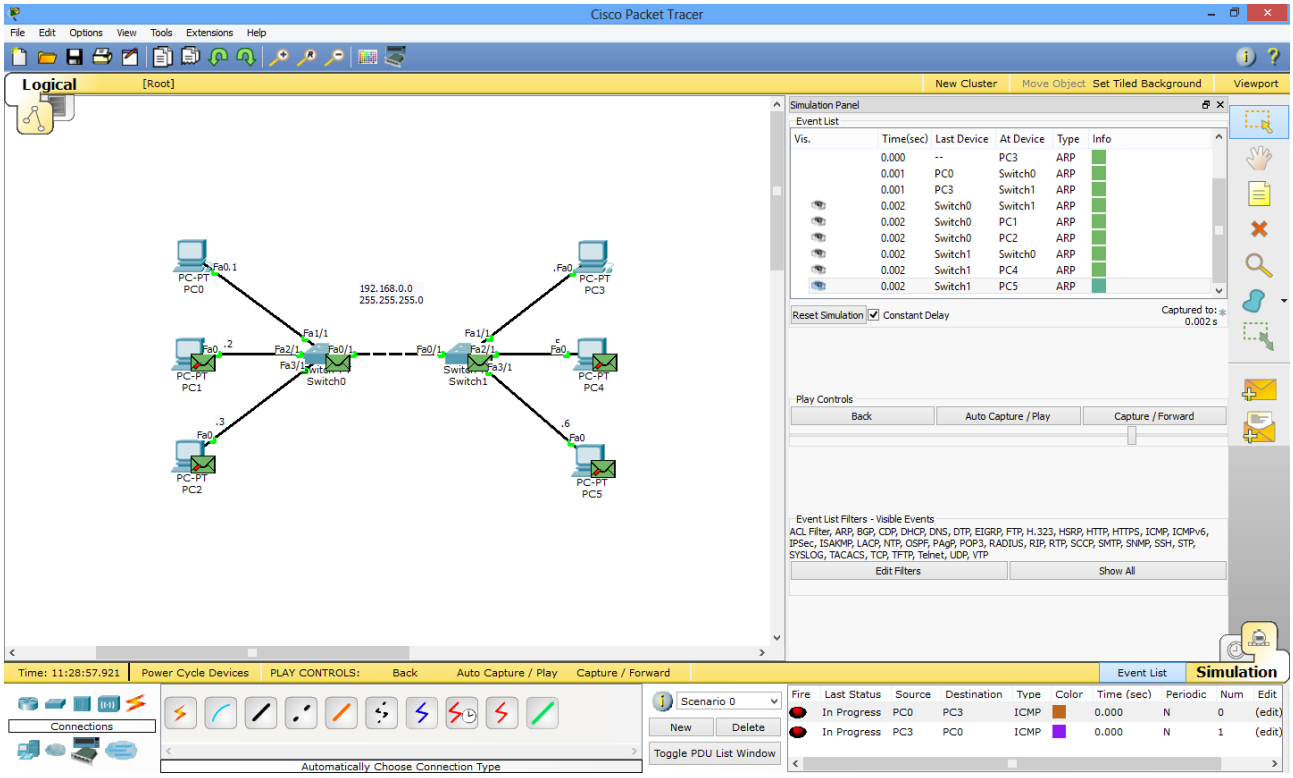
13. Now let's investigate how this functions with two Switches by replacing the second Hub. Your network should now look like below:



14. Let's repeat that same scenario once more, we'll now see how much more efficient this network is. For this exercise I will start before the ARP requests have begun to show how broadcast and unicast work on the network. Here is how the simulation looks before we start the exercise:



15. We can see four packets ready to send, two lots of ICMP and two lots of ARP. Now remembering back to earlier in this tutorial, an ARP request is a device learning where another device is. As such, the Switches also do not know, so they will send out an ARP request to build up their routing tables. We'll see it looks like this when they are doing the ARP request:



16. Notice two things here; Firstly, there was no collision! Secondly, the four other machines we aren't communicating with are showing they are ignoring the ARP. This is because they don't need to respond as the Switches are not asking for those individual computers.
17. Why was there no collision?

    A Switch has multiple collision domains. Effectively, each port is connected to another when a communication is occurring and as such each link has it's own collision domain, greatly reducing the possibility of collisions. However, if one does occur, it also has a small amount of memory on it allowing it to re-send the missing packets.

18. The rest of the simulation should simply result in the ICMP PDUs being sent, followed by acknowledgement packets being sent back. This will all occur without further broadcast ICMP traffic, bringing our network utilisation to its highest possible value.

Telnet Instructions
For a Mac you do not need to install anything or enable anything, you can simply use telnet by using Terminal or your preferred terminal application.

For Windows you will need to follow this:
> Go to Control Panel
> Click on Programs
> Click on Turn Windows Features On or Off
> Turn on Telnet Client
> Restart your machine

You will then be able to use Command Prompt to telnet. You can alternatively download TeraTerm to achieve the same result.