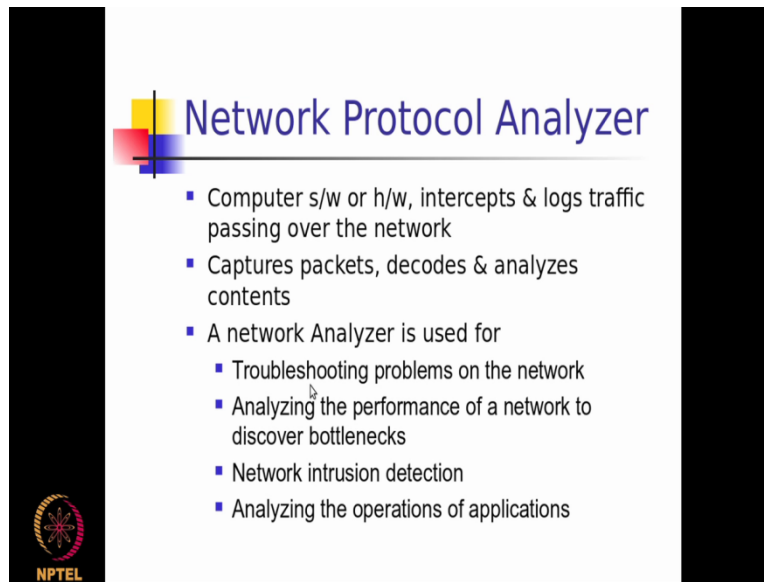




Information Security 3
Sri M J Shankar Raman,
Consultant Department of Computer Science and Engineering,
Indian Institute of Technology Madras
Module 57
Wire shark

(Refer Slide Time: 00:21)



 **Network Protocol Analyzer**

- Computer s/w or h/w, intercepts & logs traffic passing over the network
- Captures packets, decodes & analyzes contents
- A network Analyzer is used for
 - Troubleshooting problems on the network
 - Analyzing the performance of a network to discover bottlenecks
 - Network intrusion detection
 - Analyzing the operations of applications



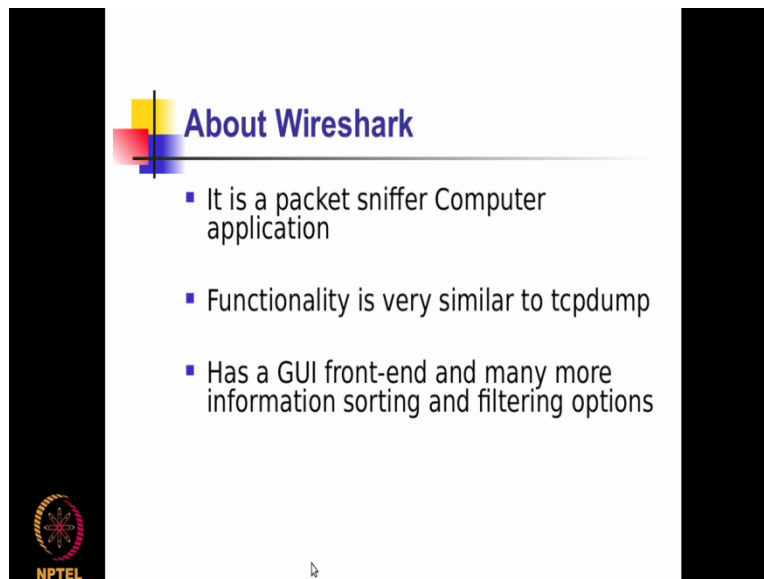
So in this module we will actually be looking at a network security tool called as a wire shark, so wire shark is basically a network protocol analyzer by which I will be able to sort of find out what kind of traffic is flowing in my network and also do some amount of troubleshooting as and when required on the network, so a network protocol analyzer is basically a way by which I could capture the packet, sort of decode the complete packet including the packet header.

The headers for the different network layers that has been added and the actual application message pay load that the particular packet is really carrying forward right, so after the decode I would also be able to analyze the contents if I have a basic understanding of that particular protocol right.

So a network analyzer is typically used for troubleshooting problems in the network, so essentially if I want to find out why certain packets are not reaching a particular server.

What kind of payload has actually come on a particular, in a particular packet all those kind of things I will be able to do an analysis with a typical network protocol analyzer, I've also be able to sort of try to find out wherever possible any kind of road blocks that is preventing a network from performing very optimally and also the network intrusion deduction part of it as well as analyzing the operations of the different network applications that could be running in my network.

(Refer Slide Time: 02:00)



The slide is titled "About Wireshark" and features a bulleted list of three points. The slide is framed by two vertical black bars on the left and right sides. In the bottom left corner, there is a logo for NPTEL (National Programme on Technology Enhanced Learning) consisting of a circular emblem with a star-like pattern and the text "NPTEL" below it.

About Wireshark

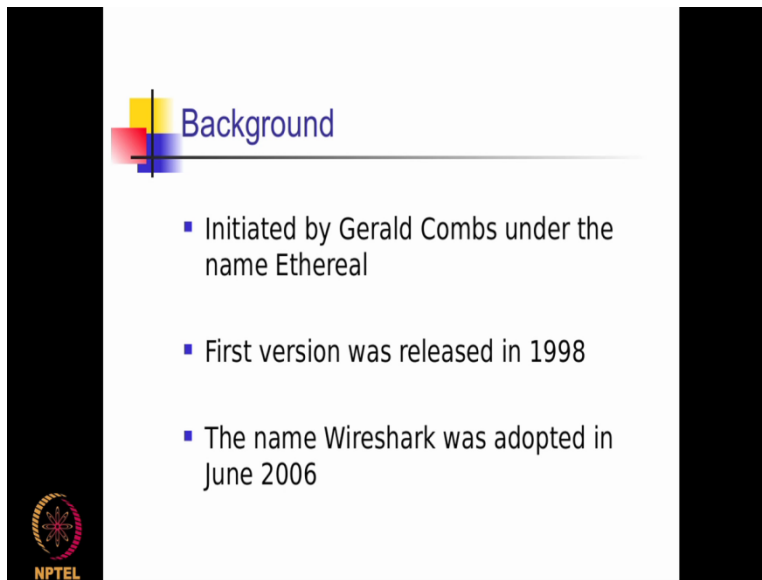
- It is a packet sniffer Computer application
- Functionality is very similar to tcpdump
- Has a GUI front-end and many more information sorting and filtering options

So about wire shark, wire shark is basically a a packet sniffer application that I could run on my on my system and with that installation of wire shark software on my system, I will be able to typically analyze what kind of traffic is actually flowing around in the network and also take a look at each of the network layer headers along with the the payload information containing the actual application message, so essentially if we've heard of something like a TCP dump application.

A TCP dump application is a command line tool that is available which will sort of take the packets coming on a particular interface and dump it in a raw manner for me to really analyze what kind of traffic is actually being flowing through the network an nab per packet bases, so the wire shark as as GUI front-end and in order to effectively do an analysis of the network traffic, it has a very powerful set of filtering options wherein I could actually filter what specific kind of

traffic. I want to be looking at among the entire set of traffic patterns that could be flowing in my in my network.

(Refer Slide Time: 03:20)



The slide features a title 'Background' in blue text, preceded by a decorative graphic of overlapping colored squares (yellow, red, blue) and a black crosshair. Below the title is a horizontal line. A bulleted list follows, detailing the product's origin and name change. In the bottom-left corner, there is a circular logo with a starburst pattern and the text 'NPTEL' underneath.

Background

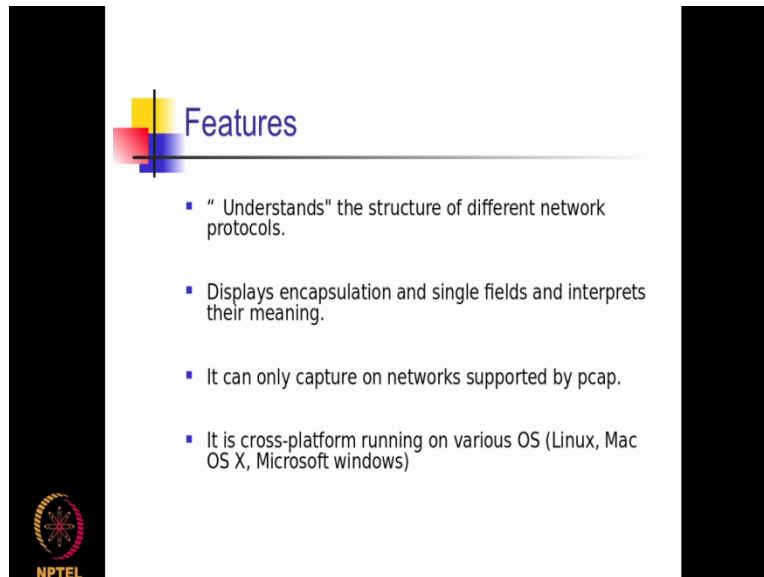
- Initiated by Gerald Combs under the name Ethereal
- First version was released in 1998
- The name Wireshark was adopted in June 2006

NPTEL

Just a quick background this was actually originally developed by engineer called Gerald combs and the old name of this wire shark product is actually named as ethereal right, so you will be able to see a lot of history of this product by searching around for the ethereal name as compare to wire shark because wire shark was just recently.

The name of wire shark was actually introduced for this product as recent as just a decade ago right? So the first version was actually released in 1998 and this has been actually very stable product and very handy tool for network administrators effectively to use for doing an analysis of the network.

(Refer Slide Time: 04:00)



So what are the features basic features this software is actually providing us is understanding the structure of the different network protocols.

So here if I am basically trying to implement a new protocol as part of a stack on a particular operating system, I could actually make use of this tool to capture those packets because when I am really implementing the the protocol as as part of the initial versions, I could always expect that I am going to be actually having bugs in my implementation and this tool because of its very effective method of capturing the traffic in a very raw manner at every bite level.

I will be able to quickly come to a conclusion on where exactly is the bug which is making my implementation non confirming to the particular protocol standard right, so in that way it helps me to understand basically the structure and the implementation of the different network protocols, it displays encapsulation and single fields and interprets their meaning, so as I was telling you right now, a packet when it is actually getting built from an application message at the higher level.

Gets transformed into a transport layer segment then gets converted into a network layer datagram and then gets converted into a data link layer frame right, at each level the packet is actually getting encapsulated with the corresponding layers header information like we've seen

in the previous module on networking and using the tool like wire shark I could actually see what are the different header level metadata that is actually been added on a per packet bases right.

And that is very very powerful whenever I am doing a sort of a conformance testing of my implementation number one, number two whenever I want to take a look at the individual level headers metadata information when a packet is actually flowing through my network right, this comes in handy both for the purpose of debugging a new protocol implementation that I might be trying to implement as part of my stack right now and also it comes in very handy for the

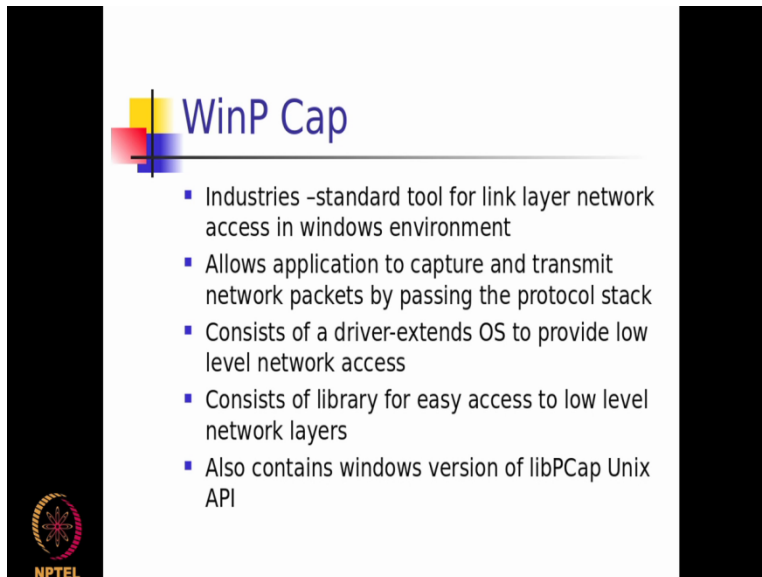
To analyze the kind of traffic that is actually flowing around in their network in order to maybe do some sort of a troubleshooting maybe with respective functionality or with respective performance right, and we also need to understand that the wire shark being a a a tool that is effectively trying to display the cap captured packets, behind the scenes it is really making use of a library called as a p cap, so p cap really stands for packet capture.

So there is a library called as a lib p cap that is actually available as part of the packet capturing facility on any OS right, so today this library is actually available almost on all the very popularly used operating systems, so effectively I need to have this p cap library installed on the system in which I am actually running the wire shark application because wire shark is pretty much a GUI front-end application which is actually giving you a front-end details.

On whatever has been the packets captured in a raw manner behind the scenes by this lib lib p cap right, so this packet capture library, so because of the fact that this lib p cap is actually been ported and been successfully on different platforms and lot of platforms like Linux, Mac, windows and so on, so I do have this also available on all kinds of a Linux flavors like solar rays hp OS X, ax as well so on all these platforms where I have the p cap library implemented.

You will find that you could effectively run the wire shark application without any problems right.

(Refer Slide Time: 07:58)

The slide features a title 'WinP Cap' with a decorative graphic of overlapping colored squares (yellow, red, blue) and a vertical line. Below the title is a bulleted list of features. In the bottom left corner, there is a circular logo with a starburst pattern and the text 'NPTEL' underneath it.

WinP Cap

- Industries -standard tool for link layer network access in windows environment
- Allows application to capture and transmit network packets by passing the protocol stack
- Consists of a driver-extends OS to provide low level network access
- Consists of library for easy access to low level network layers
- Also contains windows version of libPCap Unix API

NPTEL

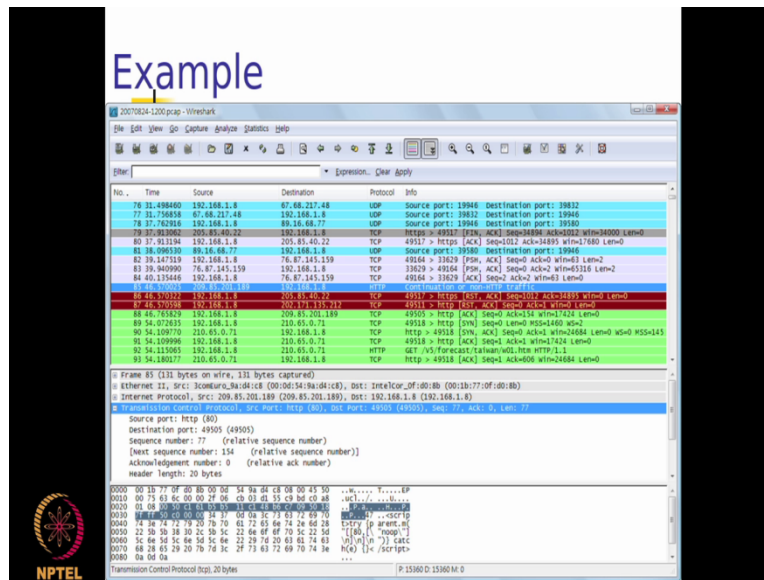
So the packet capture facility it basically a standard tool for link layer, network access in windows environment basically the win p cap is the packet capture library implementation on my windows platform and the features are pretty much similar to what is actually available on almost all the other platforms also right, so I basically allows the application to capture and transmit.

The network packets by by passing the protocol stack or also including the the protocol stack level details as well right, so it consists of a driver implementation inside the operating system on which I am really running and because of this particular implementation at a driver level wherever required I could also have access to my network packets directly at the at the lowest level, so I could even look at at a data link layer frame and then give you the contents.

Of how the frame contents really look like, as we will see in one of the screenshots later on right, so it consist of a library for easy access to low level network layers and that's basically different kinds of interfaces that I really have for capturing the packets and then sort of returning it to a front-end application like wire shark which is being done by this packet capture library so for that the front-end applications to be displaying the contents effectively right.

And this basically contains a windows version of my Unix p cap library that we were actually talking of.

(Refer Slide Time: 09:27)



So if I really run the wire shark application this is how the the user interface is going to look like right, so before we actually go into this user interface part of it, we need to understand that for the wire shark application to effectively capture the packets we did talk about from a software level the availability of the packet capture library but in addition to that the network interface.

On which I am trying to capture the packets should basically be operating in a mode called as a promiscuous mode right, now what exactly is this promiscuous mode so with the promiscuous mode the interface will be put in a mode by which it will be able to read the packets that is actually coming across in the network irrespective of only trying to find the cap packets that has been destined for it to be used right, so by putting the network interface in a promiscuous mode.

I basically get into an mechanism by which I will be able to even capture the packets that are getting into a broadcast medium and that are getting broadcasted in my in my network as compared to in a non promiscuous mode where I will be able to have the packets delivered to me or read by me only which are actually destined for me right, so in other words if the destination address of a packet for example is my ip address is my network interface ip address in a non promiscuous mode.

I will be able to read only those packets right, but whereas in a promiscuous mode if a packet is actually coming for example let say with a broadcast ip address I will be able to also capture their packet and then get those packet delivered to my higher layers in my network stack right, so one of the primary requirements apart from the support of the p cap library on the system for running wire shark effectively is that the the network interface on which I am trying to capture.

The packets for doing an analysis should be running in a promiscuous mode right, so once I basically start my wire shark application after installing it I need to basically decide and inform the wire shark application what is the network interface on on on which I want to capture the packets, so once I basically mention what interface I want to capture the packets then from that instant onwards whatever packets actually come on the network, they all those packets will be captured.

And this screen that you see in front of you right now will be getting constantly updated with the newly captured packets and then displayed right, now if you see here there are different packets that are getting listed and as I was just telling the screen will be getting continuously refreshed unless we stop the packet capture interface if he for example say that ok I would just want to take a sample of the kind of packets that has been going around.

So this sample has been captured by this tool right now for me, so I want to stop the capture right now right, so as and when we stop the packet capture the packets will be getting captured continuously and this screen will be getting continuously refresh right, so now I can basically select any packet that I want to be specifically interested right, so for example let say this is an http packet and I want to find out more details about this particular packet right.

So if I for example say that I am I I select this packet here right, so I just click on the this particular packet then in this particular frame of my window I will have the entire details of that particular packet right, so if you see here I have the transport layer protocol, so the because this is an http packet as we were discussing before in our earlier modules http is operating on top of TCP right, at the transport layer, so you have the TCP related information all displayed here right.

So you see this collapsible button so if I basically say that I want to get more details of it and press the button here I will now start seeing all the metadata information associated with this packet for the transport layer header that has been added right, so that is basically what you are seeing here like for example the source port, the destination port right, the sequence number and all this right, so all this details are basically the TCP related header details right.

And then if you see the next level you have the internet protocol, so internet protocol is basically the network layer protocol and if I want to see more details of both the metadata that has been that has been present as part of this ip ip header I could just press on this collapsible button here and like I like you are seeing the details of both the TCP related header details you will now start seeing the metadata details that has been added as part of my IP header for this particular packet.

That you have selected here right, likewise I could see at the Ethernet protocol level and then finally at the frame part of it right, so I could basically see what exactly or the bite by bite contents that has actually been captured right, so if you see here for example since this is an http package the as key representation of the payload is basically showing you some kind of html tags right, so apparently it looks like from the payload as key dump there was an html page.

That has been requested as part of this http packet and it is basically coming in as a response that is the reason why you are basically seeing different kind of html tags here like for example the slash script and those kind of things right, so likewise if I basically select on any other packet I will be able to get the corresponding details of that particular packet displayed here as part of this frame and then the payload details of that specific packet will now becoming in the last frame.

Of my window right, so this is effectively how my wire shark GUI looks like in different frames and each frame actually has different levels of details, so if I just want to know what kind of packets have been flowing my top most frame of the user interface alone gives me, if I want to know the metadata details of each and every layer header for that particular packet that I have selected in the first frame then the second frame gives me the details of the metadata.

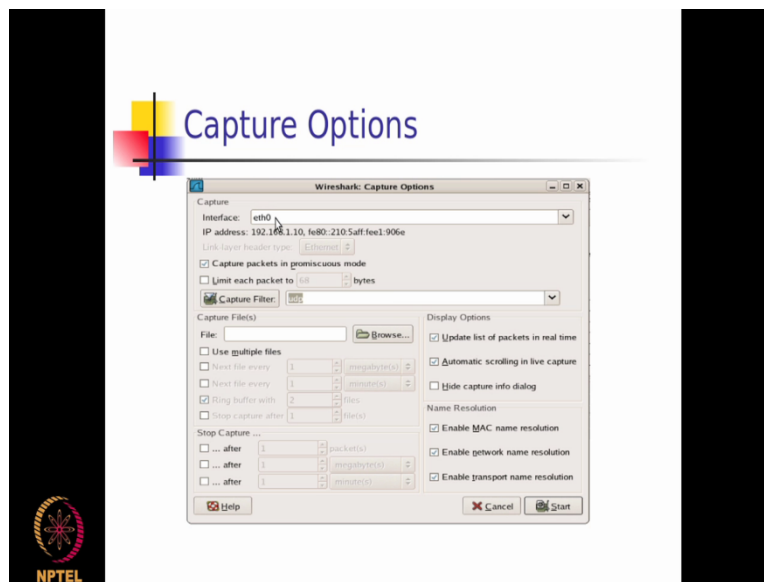
At each and every protocol header level detail, now if I want to know more index details about the real payload of that particular packet that it is carrying then I go into the third frame where I get the details on a particular payload message also right, so in that way the wire shark gives an

interface gives me the details of my traffic network traffic that is flowing around at different levels depending on what specifically I am looking at.

So if I just want to get at a ten thousand square feet level, what kind of packets are going out then my first frame of the user interface itself will give me the details right, if I want to go into the details of the header level for each and every packet that I have selected then the second print gives me those details and we want to go more induct into find out what is the payload that is actually getting carried by the selected packet that the third frame at the bottom of the screen.

Gives me those details right, so that is basically where you should be looking at depending on what is it that you are trying to gather information by running this wire shark application.

(Refer Slide Time: 17:01)



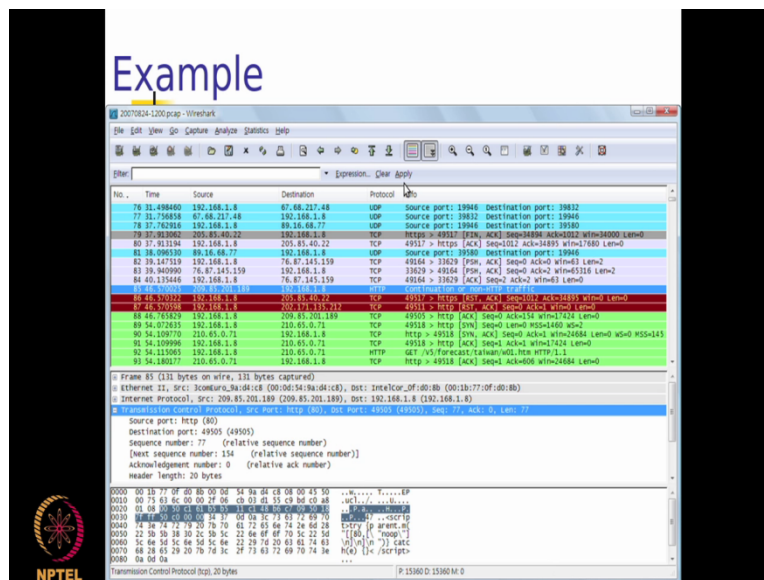
So the capture options are you have different options that you can basically set before the capture is enabled, so at the first level you can specify the interface on which that packet has to be captured right, so if it is a linux kind of a mission you will have eth zero as one of the interfaces commonly used.

So I could select ddf zero right, so whatever is a interface name that is actually configured that I want to be capturing my packets right now, I select that particular interface here and then I basically if I want to to apply some filter saying that I want to capture only let say TCP related

packets or let say I want to capture only http related packets, that filter I could actually specify here and then say start right, now what happens after this is even though there might be.

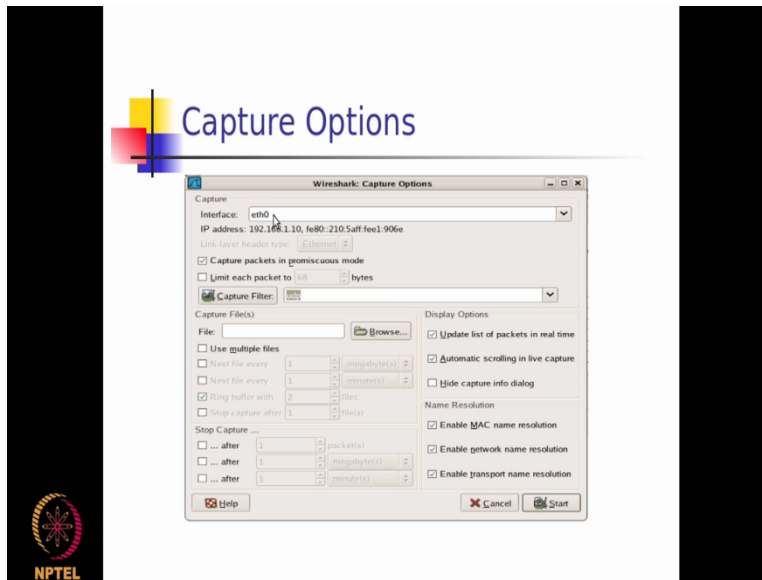
Hundred different type of packets that is flowing around in the network which is actually retrievable by my interface etf zero that I have actually selected here and let say out of that only five packets are actually http packets right, as an example because of the fact that I have selected at http or a udp here if if there are only five http packets that is that that that is going on out of those hundred packets and I've applied http as a capture filter here right.

(Refer Slide Time: 18:22)



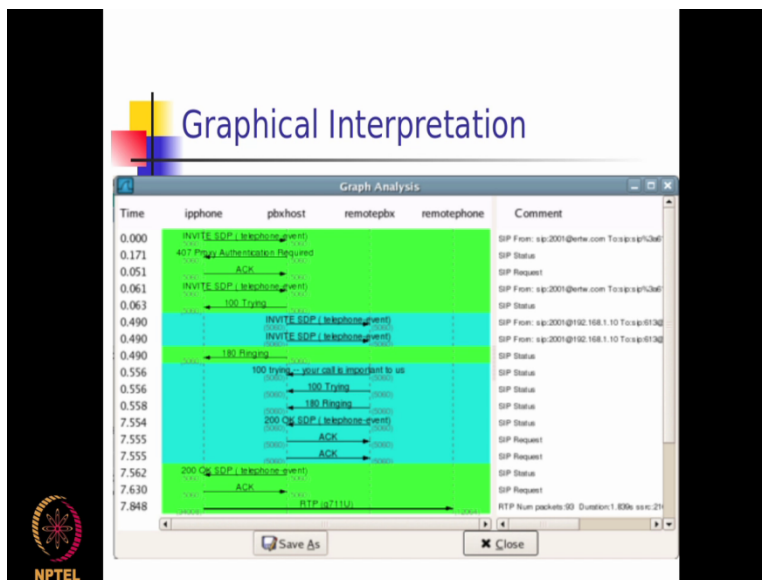
And I press start in my previous interface that I here see here with the captured packets, I will only see those five http packets alone right, so this kind of a mechanism helps me to sort of only concentrate on the find the packets that I am interested in rather than getting distracted with the kind of different packets that might be flowing around when I am not really bothered about all the different packets at that point in time right.

(Refer Slide Time: 18:46)



So this effectively helps me to basically sort of be very specific so that the clutter on the wire shark output screen is minimized by specifying exactly one filter, one type of packet alone to be captured as part of my filter, right?

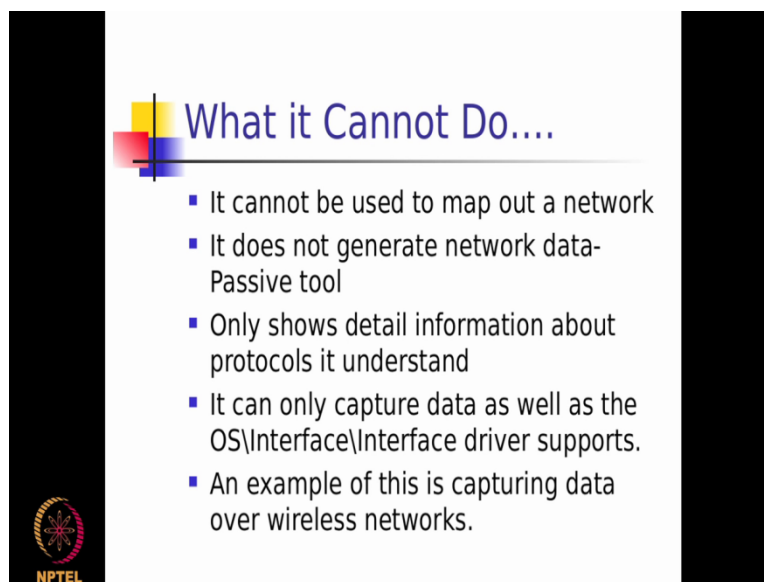
(Refer Slide Time: 19:02)



So there is also another way by which I could do a graphical interpretation with respect to the time, so it basically on a per protocol bases it also tells me from on a scale of time what kind of messages is actually been getting posted.

From one mission to another mission right, so on a per protocol bases it helps me to understand the complete message float that is actually happen between two different missions and this is the another kind of an output that is actually available in a graphical interpretation format that the wire shark produces on a per protocol bases, right?

(Refer Slide Time: 19:36)



The slide features a title 'What it Cannot Do...' in blue text, preceded by a decorative graphic of overlapping colored squares (yellow, red, blue) and a vertical line. Below the title is a bulleted list of five items. In the bottom left corner, there is a circular logo with a colorful pattern and the text 'NPTEL' underneath it.

What it Cannot Do...

- It cannot be used to map out a network
- It does not generate network data-
Passive tool
- Only shows detail information about protocols it understand
- It can only capture data as well as the OS\Interface\Interface driver supports.
- An example of this is capturing data over wireless networks.

NPTEL

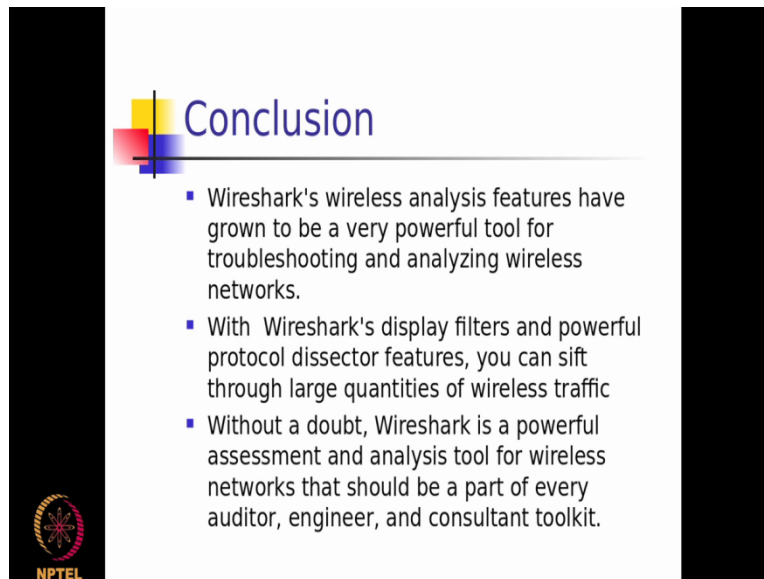
So it we will also have to understand what kind of things are not possible with this kind of a tool. This tool cannot be used to map out the network right, so I cannot really try to find out how many elements are there in my network, what are those elements, what are those IP addresses, what kind of OS is running on that and so on and so forth in this because if a system is actually up and running but it has not actually been generating any kind of a broadcast traffic for example this tool will not be able to capture that detail right.

So it is not generate network data at all, it is only a passive tool where I only get details about that the packets that are originating from other missions or from my mission without this tool itself generating any kind of a packet right, so only shows detail information about protocols it

understands so if I am basically generating my own protocol implementation right, wire shark will not be able to capture that and then give me the details of that right.

So it can only capture data as well as the the specific OS interface on which there is a driver that is actually supported for right, it can also be something which is actually used for capturing the data over the wireless networks and that is something which it can definitely do right.

(Refer Slide Time: 20:50)



The slide features a title 'Conclusion' in blue text, preceded by a decorative graphic of overlapping yellow, red, and blue squares. Below the title is a horizontal line. Three bullet points are listed in black text. In the bottom left corner, there is a circular logo with a colorful pattern and the text 'NPTEL' underneath it.

Conclusion

- Wireshark's wireless analysis features have grown to be a very powerful tool for troubleshooting and analyzing wireless networks.
- With Wireshark's display filters and powerful protocol dissector features, you can sift through large quantities of wireless traffic
- Without a doubt, Wireshark is a powerful assessment and analysis tool for wireless networks that should be a part of every auditor, engineer, and consultant toolkit.

NPTEL

So in conclusion wire sharks wireless analysis features have grown to be very powerful tool because I do have lot of security and a performance related problems whenever we are using a wire less networks.

Because inherent characteristics of a wireless network so this tool helps me to sort of iron out and sort of determine at least some of those issues very effectively right, so with the wire sharks display filters and powerful protocol dissector features especially when I want to basically understand the kind of performance related problems on protocols that wire shark understands then I will basically be able to quickly come to a conclusion by applying the filter on where exactly is a problem.

By using the wire shark tool right, so with all these kind of features it is pretty clear that wire shark is a very powerful assessing tool and also an analysis tool both for wired and wireless

networks and it it it could be used typically for doing any kind of auditing or engineering or as sort of a consultant tool that could come in handy in various kinds of scenario, so if I am really trying to do an audit of the network if I am really trying to do testing as an engineer.

Of a particular network protocol that I am implementing or I am just trying to find out for example, any kind of a performance or a security problem of all these kind of purposes tool will come in handy to give me at least the basic amount of information on the kind of traffic that is actually generated as part of the network in which this tool is right now passively listening right, so this is actually a free tool that you can actually download and install as we were discussing irrespective of whether you actually using windows OS or any kind of Linux os you will be able to download this tool and install it very easily because this is a very intuitive way of actually using the tool right from the installation to the initial setup and then the the the actual usage of it like what we were actually seeing as part of this module right.

Thank you very much.