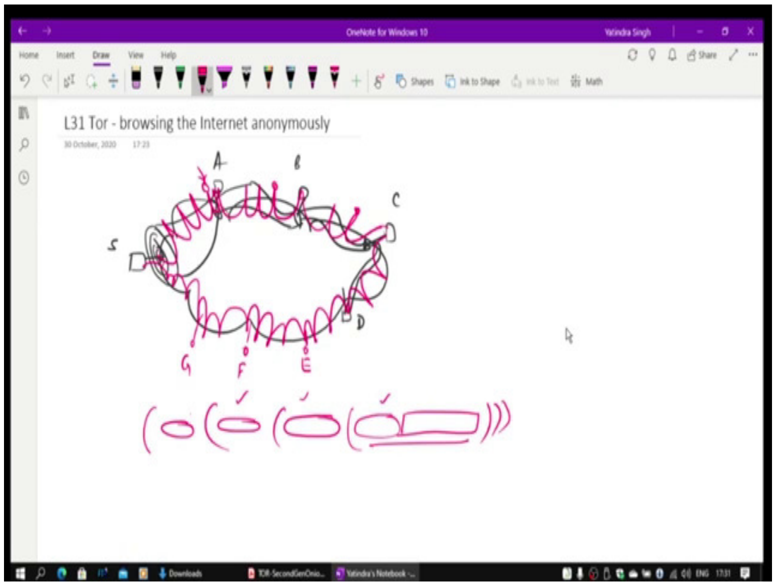
Peer To Peer Networks
Professor Y.N. Singh
Department of Electrical Engineering
Indian Institute of Technology, Kanpur
Lecture 31
The Anonymous Communication on the Internet through TOR Network

(Refer Slide Time: 00:28)


L31 Tor - browsing the Internet anonymously

In this video we will be extending from there we left in the last one. In the last video what we had seen was anonymous routing but it was the packet based system. I created on the network a DHT network is being there on internet and they are some nodes which are part of it and the strategy which we actually use there that the source knows about the destinations node ID and destination may not know the sources node ID unless the packet from source or message from source reaches to destination.

The way the communication was explained that S will now first of all encrypt the message with the so that only D can know it basically D is public key and a key which is being used for encryption under the random key that is what is being used to encrypt the message and then D is address D is node ID was being appended and this new message is now for the message which has to be interpreted by the another intermediary, For example in this case there can be intermediate can be A, B,C.

And then whatever is now has to be done by C has to be encrypted by another random key which is then further encrypted by C is public key and so on. Technically I will be create it a kind of a telescopic thing, first telescopic routing happens still A, so from there the telescope routing goes to B and from within this it goes to then C and from there the final thing was going here to the destination D.

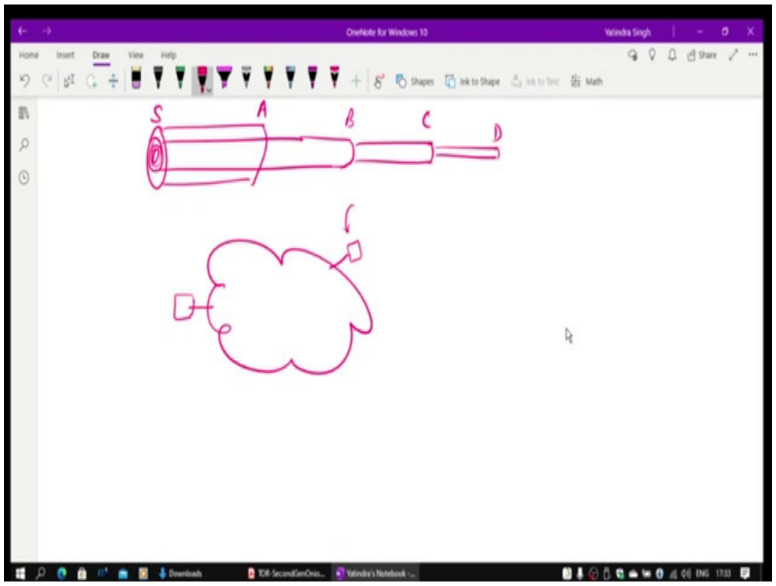
But actually this is the way it looks like but normally because the message from S to A is also going through a DHT route. This will actually take some hops from other nodes and ultimately it goes here, now A will only be knowing that it has to send a message to B, because B's node ID is going to be mention when A will decrypt the message. S information the node ID of S will not be known to A, it mean only at S will know what is the node ID of the previous node, it has to know that which are the hop sequences so it is very difficult to do it if it is being or across the globe.

From here then actually it is being going through again multiple hops because it is a DHT routing, D will only know that which is the previous hop from which the message came and it will then it will be knowing that it has to go to C. it will again go through a DHT root in this fashion and C will know only about this previous hop not about B, A or S, M knows about B, so which will then go to D.

Now, when B will encrypt D will decrypt the message it will figure out that it the message came from S. It can actually decide its own sequence of S decided for node A,B and C. It can decide for node E, F and G. Some of three other nodes in the reverse path or it can be even more and then these can be rooted back in the same fashion and to go back to the S, so that is how the communication will be happening.

This how we did anonymous communication in the previous video, important thing we actually use the message based system and there was something which was added in front of it before the encryption happens and then added something before another encryption happens, added something before another encryption happens and so on. And as you are moving forward one layer at a time is going to be stripped off ultimately the final message will go to the destination. In fact there are four layers here, so they are three nodes which will strip of 1, 2, 3 and the 4th one will be taken care of by D itself. So and this we call the telescopic routing in the earlier video.

(Refer Slide Time: 04:25)

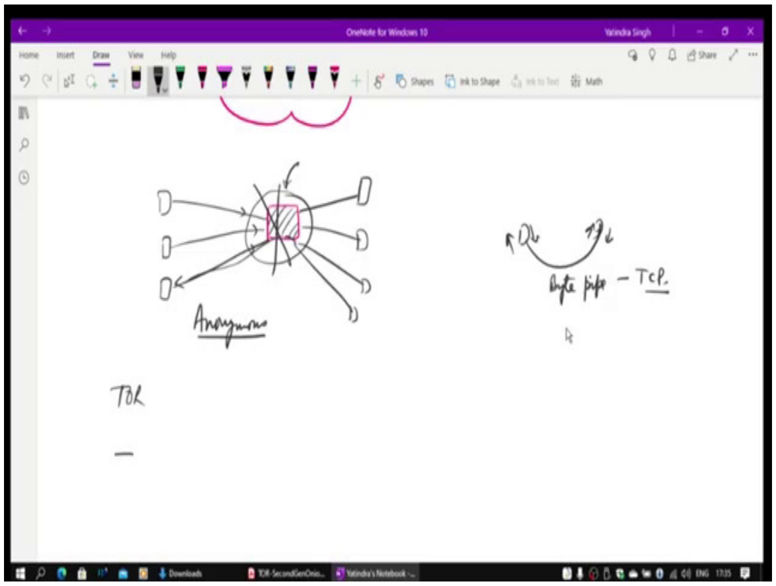


And we represent it in figuratively in this fashion, this is going to be source this is going to be A which will do the first decryption and then there is a packet which been passed on to B which will do another decryption actually here. C,which will then further to a decryption and ultimately D which is going to do the final decryption actually. So, this was the telescopic thing which was done, but this was okay if when it is a messaging based system.

Now, what I want is and remembers both S and D were actually the nodes which are participating in the DHT network that is important, but now what I want is. I want anonymous browsing. You are a browser you want to browse certain websites, you do not want website to know that you are what is your IP address and port number, how to do this actually? So, one of the common ways essentially basically anonymity to the browser, but you know whom you are browsing.

And this browser in our normal internet this is not in your DHT Network. So, how to do it? So, one of the simplest way of doing it is actually using an anonymizer.

(Refer Slide Time: 05:41)



You can actually have an anonymizer server which is running and then everybody then wants to communicate to some server has to talk to this anonymizer server and this on behalf of the client will now go and attach connect to the server actually and we will feed the information and we will send it back to the person who has requested.

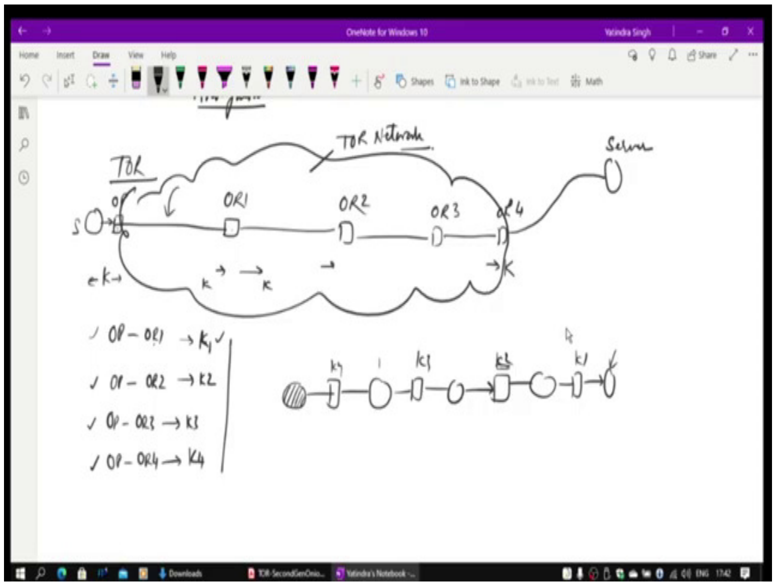
Now the problem is this guy knows everything, so normally in any anonymous system even intermediary should not know what is happening. But the way it happens is we normally create HTTPS connection between these two, this will act like a proxy and once the secure connection is done they will just communicate, so only these two endpoints will know this guy will not know what communication is happening.

But it knows that this server is communicating to this particular client that information is available and secondly if this guy dies off then ofcourse there is a problem with this crashes that is it, the connection will not be happening. This anonymizing servers are okay solution which people tried but these are not reliable. Secondly it is a server somebody who owns it will have access to everything. There are single points where the compromise actually can be made.

This becomes very lucrative even to destroy so that it coming whole communication can be destroyed, you wants something more rugged. So, what we want is we want at the tor actually works in this form and remember the connections which you are making I am talking about the circuits or TCP connection, you set up the connection, connection is maintained then you keep on sending bytes, bytes will go all the way to this guy in the reverse path it puts bytes, bytes will be rotating comes back to this person. It is not a separately every time a message is going, of course in the underlying network layer it will be happening.

But as far as the applications are concerned for them it is a byte pipe. So, if basically two applications are connected through a path you push in the bytes, you kept bytes will come out from other side pushing the bytes, bytes will come on this side. Essentially this is what we are trying to create and this is what is done by TCP transport control protocol.

(Refer Slide Time: 07:53)



Now what we require is a very interesting thing I need a byte pipe, so this has to be encrypted by something and this goes to a node, this can be further encrypted. We needed something a very different system here, where if then number of K number of bytes are sent by this guy, the K have received here and K are further transmitted, K are received K are transmitted and this happens over multiple hops. Now, can I do telescopic routing here?

So, ultimately at the end also I will still receive K and there is also still a multiple layer of encryption which is maintained, so that is one thing which we need to do. So, how we can do this actually? Tor works on this particular thing is basically still works for TCP circuits it creates circuits from on the tor relays, so we call them onion relays (ORs).

The applications this will be also OR and this is exit nodes so this can now set up a TCP connection to a server in the internet and the application here we call it a onion proxy. So application will be running here, application will have onion proxy running which essentially makes it gives an interface to the application to connect to the onion network and this is what is going to be the basic structure which will be available and we call this network as onion networks or tor network, the onion router network.

So, each one is a router and we create what we call circuits, so there will be a TCP if you put the bytes, bytes will just go as it is and bytes will be removed. We will now set up something kind of a virtual circuit, so there is a circuit which will be there and these tor relay or tor routers or onion routers will be used to create this anonymity. How the anonymity done is an interesting thing.

Now, remember it is not going to be like the way we did it is not going to be through a DHT routing going here. This guy knows onion proxy knows about this onion routers IP address and port number it communicates, but it is actually done after doing encryption. So, this routing is not there, so I can actually remove it, so this is direct TCP connection, this onion router will connect to this, this onion router will connect to this and each onion router will receive the message, OR will receive the stream, we will decrypt it and further forward it and it has to figure out whom to forward there is going to be mechanism here for doing that.

Now, the way it is telescopic thing will be done is this onion proxy has to create has to have some mechanism by it will set up a encryption key with this onion router. So, onion proxy to onion router1. I can write them 1, 2 and 3 and 4 there will be one encryption key form created with this I can call it key1, onion proxy will also create a key with onion router 2, I call it key 2, onion proxy will create with something with onion router3 which is going to be key 3. These keys they have to be established.

Now, important thing I will take the K bits where they typically use mechanism encrypt certain payload or certain number of bytes, so little bit it and done encryption with key1. Number of bytes will still remain the same. So, this onion proxy will do first of all encryption with key 4, key 3, then key 2 and key 1, it is one after another, you would kind of process every time there is a encryption with key 4 whatever is the output further encrypted key 3 actually now and then key 2 and then key 1 and this is what is going to be transported actually.

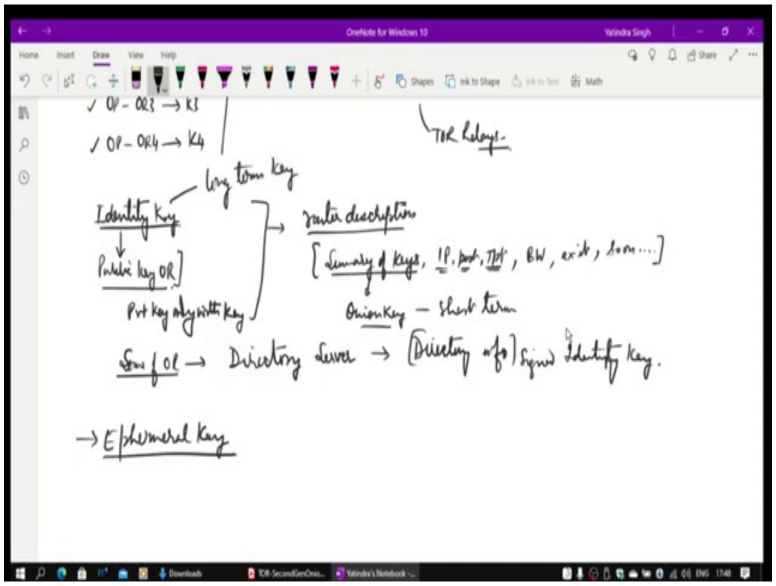
So, once it come to OR 1 it will decrypt using key 1, it will get this, this will be transported to onion router 2, which we know about this key 2, so remember different onion routers knows different key, so they will be keep on encrypting so ultimately onion router 4 will actually have this particular message. So, that is how the onion routing will be happening and once these paths till the exit node we call it exit router, is known. Application and talk to him and to act like a proxy to set up a connection to a server.

This guy will set up so server will know only its IP address and it will set up as TCP circuit between these and S will be actually sending for example its security certificate will be doing transport layer, security. So that will all these bytes will be just transparently transported back to this application, so they will be another pipe encrypted pipe which will get created all the way to the server which is using TLS.

Now, of course if this application or source wants to identify itself through server it can do that, then only server will know who is the source is, may not be knowing its IP address and this guy explicit tends but user ID and other information can be found actually.. But not the IP address and port, so this will be acting like a proxy. Now, these circuits we can keep on dynamically changing this onion process should have capability to keep on setting up new ones and keep on sending the circuits.

Those are server now, I have to be as exile so, that it can now connect to another IP address and port number. So, far if it is identifying this circuit by this IP address and port number this circuit has to be maintained. So, each onion router will receive will decrypt and forward that is what is going to happen. Now, each onion router will actually have something called identity.

(Refer Slide Time: 13:54)



So, we need to define an identity, so we call it identity key, the basically is a public-private key peer so the public key will be known to the whole world, then that is what public key of onion router is what is going to be the identity key. The private is only known to the onion router, so private keys only with the onion router the corresponding one, nobody else knows.

Now, this identity key is going to be used for certain purposes, so it is basically going to be used to essentially create a router description, so each router onion router will have a description. So, this router description will normally will contain a summary of keys. These summaries of these are basically onion keys, onion keys are also public-private key pairs.

Private key will be with the router, so they can this can be modified periodically and this information is deposited with the directories, directories servers, so some of these onion routers will act like directory servers also. So, summary of keys which will be maintained by them, it will talk about what is a IP address, port number, transport, so normally it is a TCP transport which is always used in this case because it is actually built for TCP.

And then it will also contain the bandwidth which is available because you might actually get this router description from a directory server and then you have to find out the guys which are having good bandwidth and only use them as your tor relays the subject you are going to set up through those tor relay, these are also called tor relays I mean routers call them tor relays.

Bandwidth will also be mentioned, there will be exit policy, so for example if I am running an onion relay in India. I do not to want to actually communicate to any server in Pakistan. I will say if you are going to ask me to set up a stream or a TCP connection to some server there I will deny, so that is possible. If there is a exit policy and then other information which can be there and it is not been specified anything, this description will be digitally signed by the corresponding private key, this identity key and identity key can be used to verify this one.

Each router will essentially be identified by this key that is a node ID for each router, now some of these ORs, some of these onion routers will also act like a directory server. So, why I need a right directory server here? Now, if you look at in a DHT network, the way things happen is I actually maintain a routing table there is no routing table here, we have to connect to some directory server and from there each directory server will be essentially exchanging their information with other directory servers and keep updated structure.

If I know one directory server, I will be also maintaining some directory servers like a unstructured thing and I can fetch from directory server the list of all onion routers which I would like to use. Now, this directory information, the directories which will be created by something which also will be signed by the node ID. When I am fetching a directory from directory server or directory information this will be signed by directory server, which will then contain a list of routers and then their outer descriptions I can find.

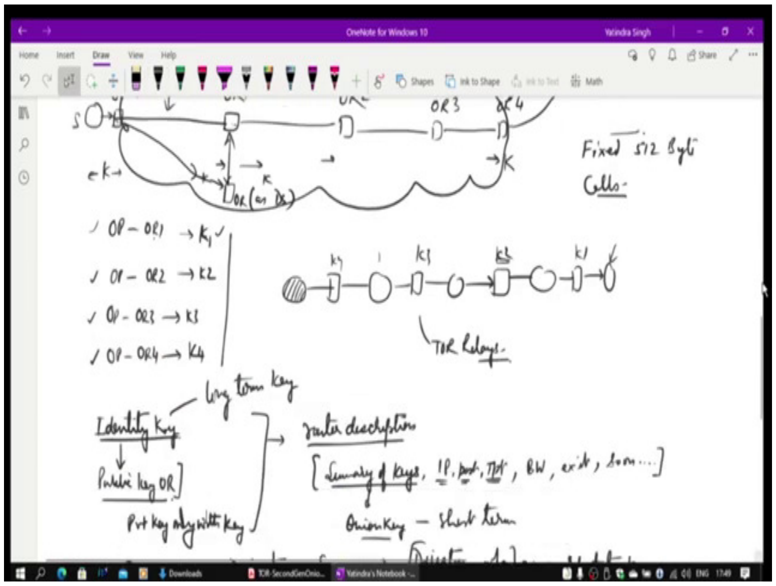
And router descriptions will be signed by their corresponding private keys, so I know this node ID will have this information and is a private key already moved into this must be the original guy, who must have created this description that is not a fraudulent person. So, directory server's, directory information will be always signed by the identity server. So, this will also be signed by identity key of the corresponding node who is acting as a directory.

Now, there will be now maintaining these, what we call onion keys, these are again private key things. These are basically used when you want to set up a connection with circuit with a certain onion router, you are not going to use it identity key to send random numbers so that or basically creating a for that you will not be using this particular key peer you will be using the onion keys which are available here and they can be very if you can be if you currently change, this is the long-term, long term key actually.

And these are short-term keys, short duration keys. And still short duration keys will be the one which will be the ephemeral keys which are going to be created using these onion keys, the ephemeral keys will only be for that in session, these are very short keys we have just been created and these are mostly will be a symmetric keys. These are essentially being created for end-to-end encryption between two onion routers.

Mostly TLS will set up the ephemeral keys, you make sure that you always maintain a forward secrecy. So, if somebody something gets compromised the earlier all communication sessions cannot be still decrypting it, because they actually were used using ephemeral keys and this ephemeral key cannot be generated, it is a randomly generated stuff. But it is only known to people when they communication happen. Now, how this two onion routers here we are going to talk. So, let us look at this.

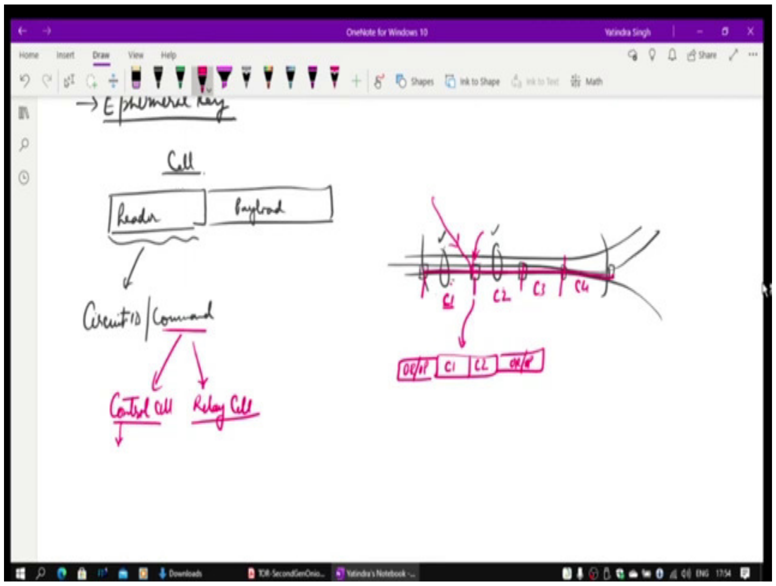
(Refer Slide Time: 19:48)



In this case we normally will have something called cells, cells will be going from one node to another and from this to this from this to this. There is going to be fixed 512 byte cells which are going to be used for communication it all circuit information will be passing through this. So, this communication between OR and OP and OR will be happening through cells, so in fact when you are going to talk to two OR’s will be talking they will be talking through cells.

You might be talking to another OR which is a directory server. So that can be some OR which is acting as a directory server. So, OP will be talking to this also through cells, this will also be talking through these cells. So, for passing on the cells first of all they have to create ephemeral keys using onion keys and then they will create a path and then on top of on that actually they will be talking to each other. Now, these cells will have a structure.

(Refer Slide Time: 21:05)



So, these cells 512 byte cells will typically will have a header and they will also have a payload, the two parts which are always maintained in this, so that is the cell is. Now, this header part will always contain two things, one is a circuit ID, Now, this is the circuit which we have been which will identify a path from one onion router to exit onion router. I can actually create multiple circuits and each circuit will be uniquely identifiable and there is one to one mapping between the previous hop and the next hop.

It will contain circuit ID and second thing will be the command and depending on this command and it is possible that I create a circuit from one onion router to through many of intermediate ones one exit router. And now there are many TCP connections which are going out from here. The onion part, I am using a still the same circuit. I will use same circuit ID all through and not all through is basically on this thing for all these flows there is one circuit ID on these flows there is one circuit ID, but this circuit ID and this circuit ID may be different actually.

But once the circuit is from source two from this source to this I have created a circuit and these TCP flows are being multiplexed through this one single circuit. That is this is a one single circuit which is this one single TLS connection actually, which has been created and this is done through again telescopic encryption which I will explain further. Now, this circuit ID is going to be connections. For example here to here there is circuit ID, I give some value say A1, sorry C1 I can call it.

The next one need not be C1. I can now create a C2 here whatever then used but then this router will now maintain a table which will say from this hop in my circuit ID C1 has to go to this particular onion router and this will be converted to C2, it will also maintain what is a previous OR and what is the next OR, OR OP whatever it is.

Let me keep on maintaining so OR OP means it knows the IP address and port number here IP address and port number and this circuit to be said this like the way you do in any virtual circuit switching system the same functionality is being used by this onion router. So, this circuit ID will be based on whatever is available. So, whatever is available they will be picked up, but only one circuit from here can be mapped to C2, it is not that you coming from another path that is also mapped to C2 that will not be happening.

That is not permitted it has to be another C2 because that is another independent circuit till another onion router. So, now based on the command part there will be two kind of cells which can be there, one of them is called control cell and other one is called relay cell and this control cell will always be interpreted by the node which is receiving it. This will always be interpreted by the node which is receiving it.

If I am sending control cell from here to here and on this circuit C1 ID. This will be interpreted by this guy, I have already set up a circuit from this point to this point C1 to C2, C2 to C3, C3 to C4 to this, but the control cells sent in from here to here will be always interpreted at this node. But if it is a relay cell, relay cell will simply will be looked at this flow table, we will keep on doing it till it reached to the last node and there is a way to verify that now the circuit is closing and then that that point of time the node will interpret what is the information know that node in between will actually figure out what are the header values in the relay cell.

I am actually stopping here and in the next video I will again start and we will talk about the various control commands and various relay commands which are used in the relay cell and we will look into details how the connection circuit is going to be set up, how circuit can be extended, so that we will be doing in the next video.