Computer Networks and Internet Protocol
Prof. Sandip Chakraborty
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

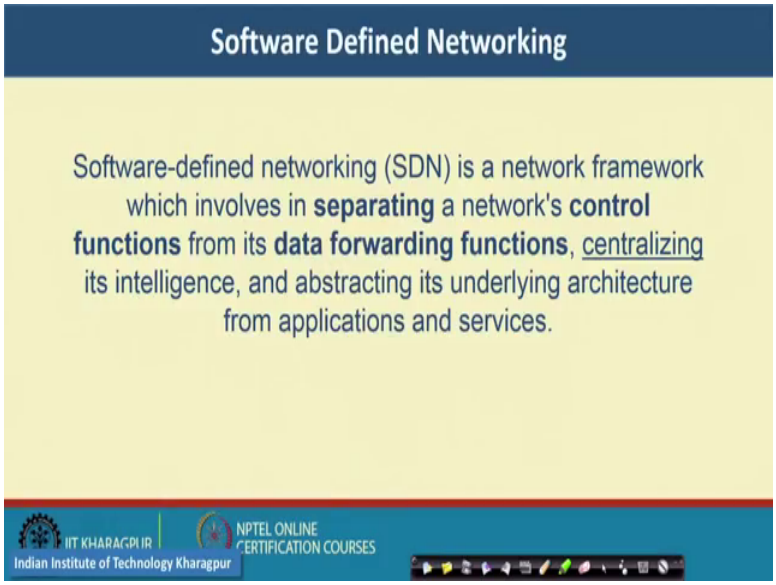
Lecture - 43
Software Defined Networking – I (Basics)

Welcome back to the course on Computer Networks and Internet Protocols. So, till the last class we have looked into the detailed design of IP routing mechanism and the structure of IP router. Now we will go to a little advanced topic which we call as the Software Defined Networking.

So, this concept of software defined networking is an recent and upcoming standard and all the traditional routers are expected to be replaced by SDN enabled router. So, we will briefly discuss about what is SDN? What is the utility of SDN? How SDN differs from a traditional router architecture that we have discussed earlier.

And then we will look into certain topics in SDN that how will can program a router with the concept of SDN technology. And how we are gradually migrating from a traditional distributed router architecture to the SDN supported router architecture. So, the concept of SDN is something like this.

(Refer Slide Time: 01:27)



A software defined networking architecture it is a network framework which involves in separating a network's control function from its data forwarding function, and centralizing its intelligence, and abstracting its underlying architecture from applications and services. So, that is the kind of formal definition of software defined networking. Now the broad keywords inside these definitions are as follows, first of all we are trying to separate out the control functionalities and the data functionalities inside a router. What is mean by separating out the control functionalities and data functionalities?

So, in the last few lectures we have seen that well inside a router you have two different levels of abstraction. You have the control plane which is implemented as a part of the software which implements the routing functionalities and the construction of the routing table and its management, and we have our data functionalities.

In the data functionalities your task is to forward a packet by looking into the destination IP filled in the IP header, and making a match with the routing table the local copy of the routing table inside a interface that is the forwarding information base, and then forward the packet to the outgoing interface. Now, these control functionalities and the data functionalities traditionally they are implemented in a single router. Now whenever you are implementing the control functionalities and the data functionalities in a single router, then the complexity of the control functionality becomes higher. Why it becomes higher?

Because now you have multiple routers with their control planes and those control planes need to coordinate with each other to generate the global routing table, or to manage the global routing table. And, these control need to be performed in a distributed way because of its architectural limitation at the way we have designed this traditional router architecture.

And with this distributed control architecture first of all your routing protocol gets problematic as we have seen that both distance vector and the link state routings have significant limitation in terms of their scalability, distance vector routing cannot get scalable because of this count to infinity problem, where as the link state routing protocol that cannot get scalable. Because of its size of the link state packets or the size of the link state information that you need to maintain if you implement it over a large network.

So, because of such limitations we have restricted this link state routing and the distance vector routing within a local internet, or within a subnet. And from network to network

we have this border gateway protocol which implements the policy. Now this may difficult for the network managers because, if there is a policy change then you need to update every individual router. And, all the routing protocols in all the routers control plane they need to get coordinated with each other to make an policy update at the individual routers, and obviously, in a distributed architecture it will take time.

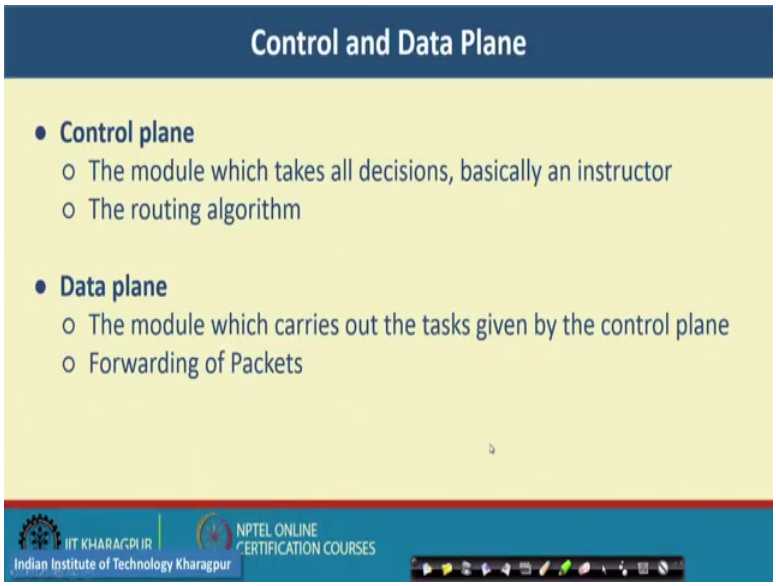
And because of this time requirement there can be inconsistencies across the routers and these inconsistencies can get significant in a large network. So, that is why managing a router, managing a subnet with some say 1000 routers is a very difficult task. And you are deploying these routers not in a single day gradually you are expanding your network and you do not know that what was the configuration of the earlier routers and you need to make a match of the configuration from these two different routers.

Then comes off the compatibility among the vendors, it is not like that all the routers of an organization they will come from Cisco, or even if they come from Cisco they will have the same model as you make a gradual deployment that outers may come from different vendors. The routers may have different models their configuration options may be different if you just look into the Cisco IOS manual you will see that it is a some 5000 page document. So, the management functionalities are very complex.

And with this distributed architecture, maintaining consistency across the configuration of the routers at the control planes are different level did that become difficult and that is why we gradually try to move from a distributed control plane architecture to a centralized control plane architecture. And, that is the basic motivation behind the design of a software defined networking concept.

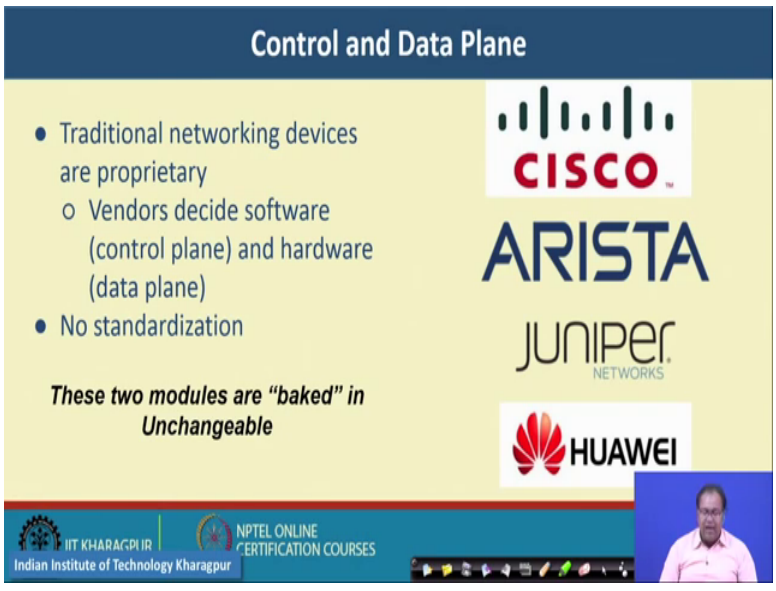
So, let us go to a little details about this SDN abstraction here the idea is that you separate out these control planes from the routers and make a centralized control plane. So, you take out the brains from the router. So, this control planes you can say that it work like a brain of the router because, it makes the decisions and that he came hard were just making a forwarding processing. So, you are taking this brains out of this individual routers and putting a centralized, putting the brains in a centralized place which is your entire route controller.

(Refer Slide Time: 07:19)



So, as you have looked earlier that the control plane and the data plane. The control plane is the module which takes all the decisions; basically it is an instructor the routing algorithms implemented in the control plane. And the data plane is the module which carries out the tasks given by the control plane the forwarding on the packets.

(Refer Slide Time: 07:40)

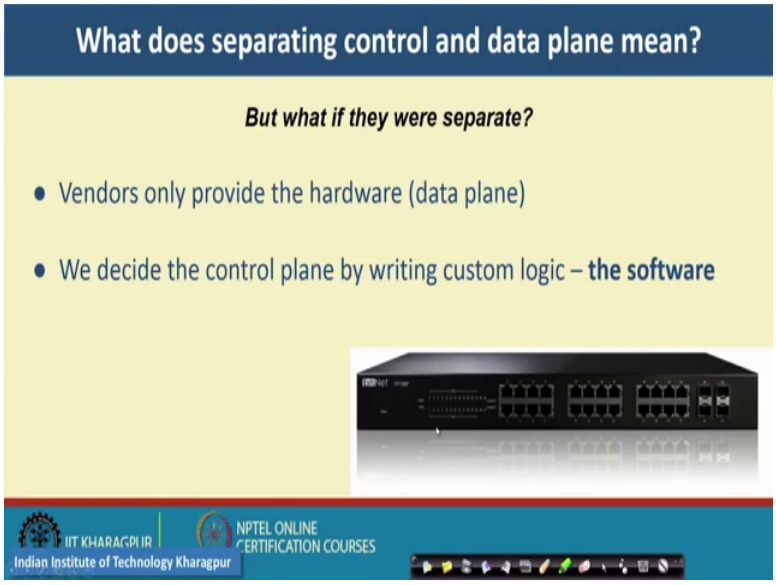


Now, the traditional networking devices they are proprietary. The vendors they decide the software and the hardware both the control plane and the data plane and there is no such standardization that there should be this kind of match. Every vendor apply their

own optimization. And because of that it is very difficult to purchase the hardware from Cisco, and then take another operating system and load it on a Cisco router. Although, there are certain routers which can support open source network IOS or router OS.

But they also have their own restrictions in terms of performance and manageability. But for the commercial routers in general the hardware and the software both comes from the same vendor and it is difficult for interoperability, managing interoperable. Interoperability is possible, but managing interoperability among products from different vendors as the kind of difficulty in a large network.

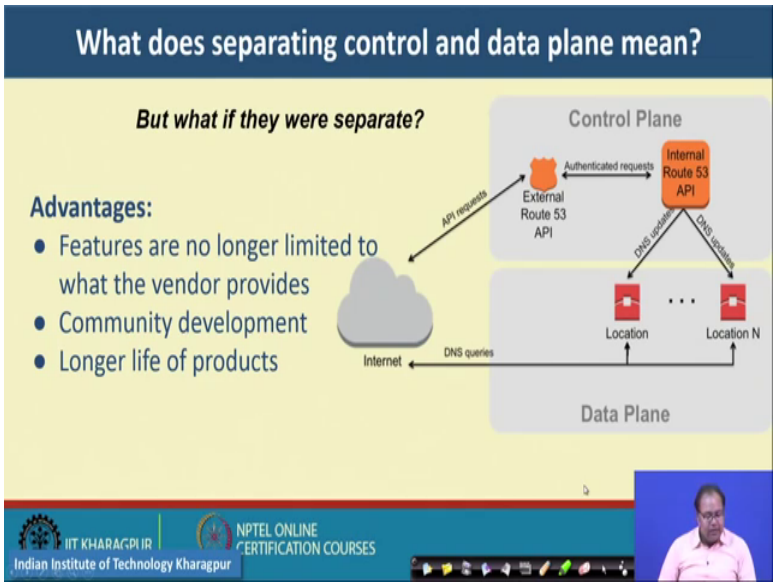
(Refer Slide Time: 08:47)



So, the idea is to separating out the control plane and the data plane. So, the idea is that the vendor will only provide the hardware that is the data plane and we decide the control plane by writing the custom logic that is the software.

So, the control plane will we decided by the application designer, or the network manager or the network support team whereas, the data plane will only come from the vendors. So, not now the vendors they will just deliver a dumb switch it just have thee TCAM, just has the TCAM hardware along with the forwarding engine, the control logic is not there. We will implement our control logic ourselves.

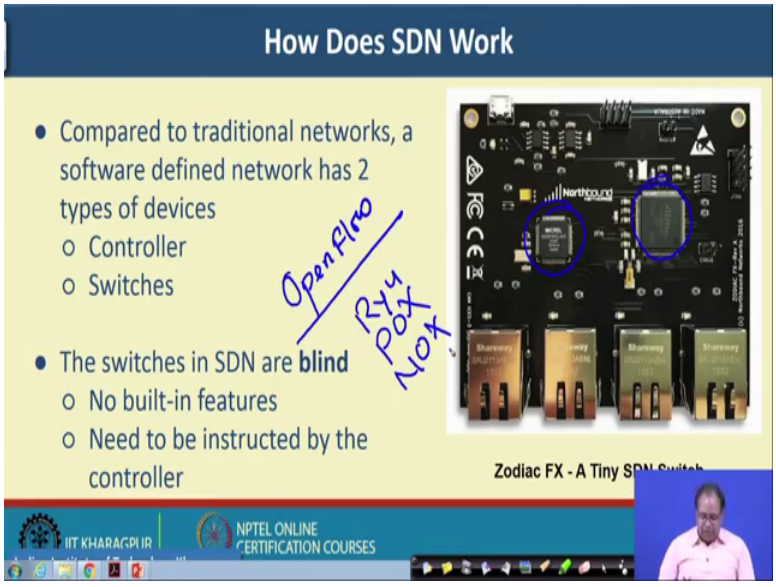
(Refer Slide Time: 09:30)



So, the vendors will only provide the hardware and we will decide the control plane by writing custom logic. The advantage is that first of all the features are no longer limited to what the vendor provides. You can always write your own network application as a part of the controller.

Or the community development in our open source movement people can come together and design a new network protocol and implement it on a control plane itself. And you do not require a vendor support for that and it; obviously, increases the product lifetime.

(Refer Slide Time: 10:13)



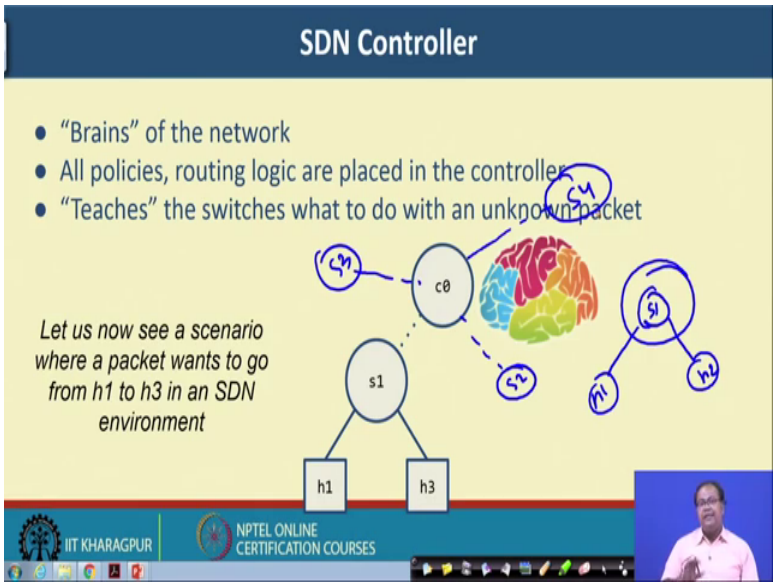
So, here is a brief idea about how does SDN work? So, compared to the traditional network a software defined network has two type of devices. The controller which is the brain of the network and the switches that is the hardware devices they are kind of dumb switches that they do not have any logic in built inside them. So, the switches in SDN are kind of blind switches. So, they do not have any built in features and that needs to be instructed by the controller. So, the switches so here is an example of an SDN switch zodiac effects switch which is a tiny SDN switch, it has 4 interfaces and TCAM hardware.

So, this is the TCAM hardware and this is the microcontroller; microcontroller for the switch. So, it just comes with this much of hardware and whatever routing logic that would be there that will be instructed by the controller itself, and the controller can comes from different open source standard. For SDN we have this protocol called open flow. Open flow is a open source standard for making controller to switch communication and based on this open flow standard there are multiple open source controllers which are available there is controller like Ryu, and many others SDN controller like the old controller was something called POX which is a Python based controller then NOX.

So, this kind of controller are there open daylight the I can name a few other controllers. So, there are difference at open source controller whatever controller you prefer you can use it in a standard computer. Now this entire brain can be put on a standard computer you do not require specialized hardware for that because route processor is nothing, but a general purpose processor. So, that is why you can put this entire control logic on a single computer.

So, you can install this controller one of the controller Ryu, POX, NOX open daylight anything whatever your is your personal choice. You can install it on a personal computer and from that personal computer you can make the things communicate with each other.

(Refer Slide Time: 12:36)



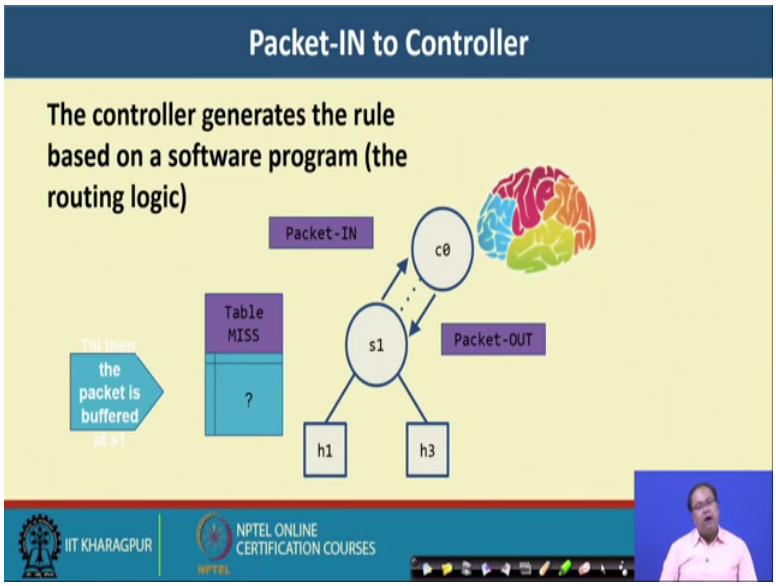
So, this is the architecture the call we have a controller which is nothing, but computer general purpose computer that works like the brain of the internet work and then we can have multiple switches. These switches are the kind of dumb polycystic. Controller actually decide and teaches the switches how to forward a packet and then you can have multiple hosts.

Now let us so this is a very simplified architecture I am trying to explain you the basic concept with this simplified architecture. So, let us look into an example that how the entire thing works in an SDN environment. So, in a traditional networking environment you do not have this controller.

So, you only have this switch and the host, and the switch has to center routing logic. Now here the routing logic is taken out from the switch an it is put on controller. Now note that you can have multiple switches which are connected to this controller. Indeed term all the switches in our organization they can be connected to a single controller.

The controller will actually perform this routing logic in a centralized to it that way we are actually avoiding the problems associated with a distributed routing logic and we are also reducing the overhead which comes from the distributed routing protocols. And we are putting this entire information in a controller which will dynamically teach the switches about how to forward the packet. So, let us look into the example.

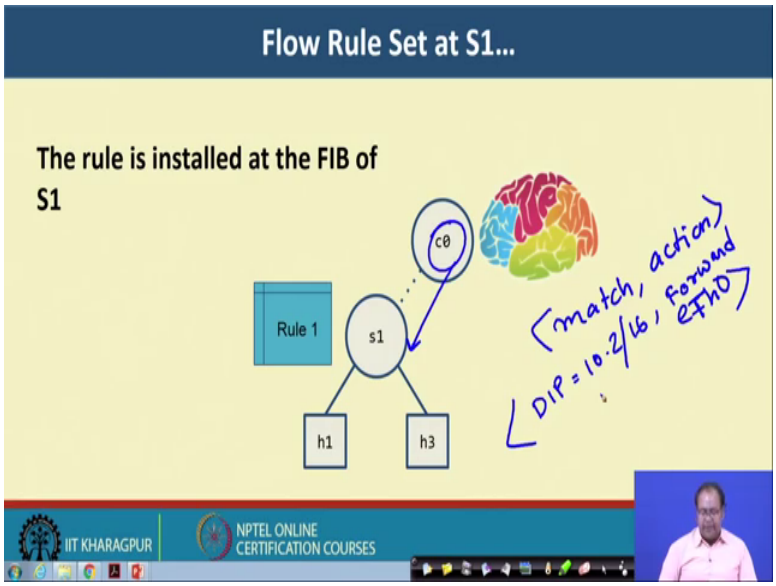
(Refer Slide Time: 14:17)



So, you want to forward a packet from h 1 to h 3, your source is h 1 under destination is h 3. So, the host forwards the packet to switch s 1. Now whenever the packet comes to switch s 1 initially this switch does not have any information. It just have a TCAM hardware, and switch fabric, so it does not know how to forward the packet.

So, what the switch does? The switch sent an packet in even to the controller; that means, the switch informs the controller that I have deceived a packet. With this packet in message, it sends the packet information the packet metadata to the controller. And then the controller actually decides that what to do with that packet and return back the information to the switch in a packet out event. And till that time the packet is buffered at s 1, buffered at the switch.

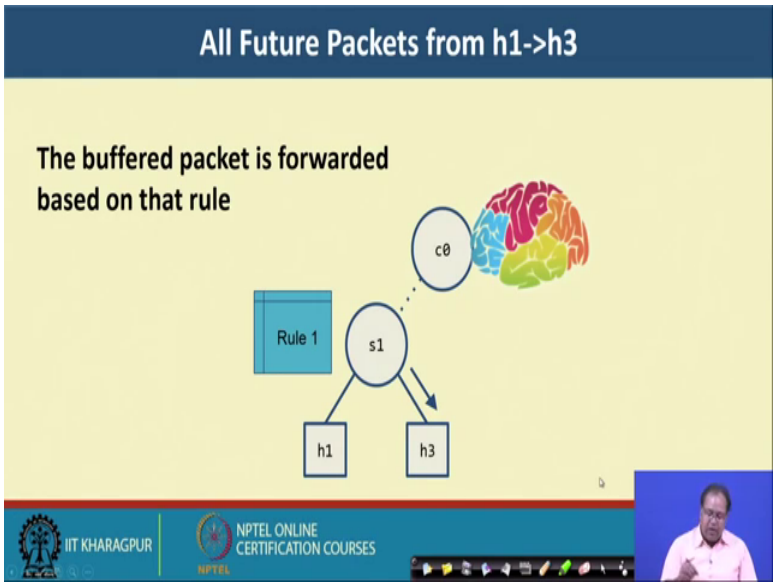
(Refer Slide Time: 15:16)



Now, the controller sends to rule to the switch then this rule is installed in that TCAM hardware of the switch. So, well discuss the open flow protocol in details in the next class during that time I will show you that how we actually write the rules and how a rule looks like. And this rule is actually a very simple thing the rule is just kind of match action pair right. So, a suit rule is nothing, but you have a match data and then action data.

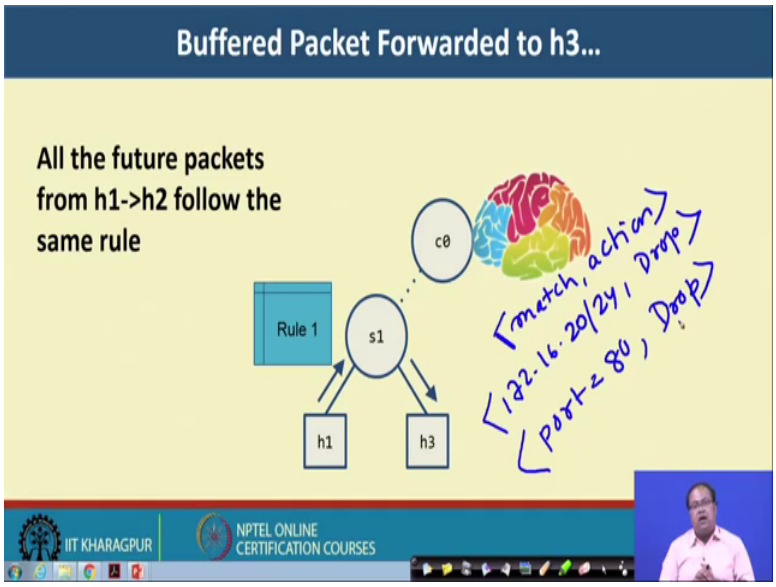
So, the match data says that say if your destination IP is some 10 dot 2 slash 16 then you exceed any reaction is say forward, forward to say interface eth0 forward it to interface eth0, so that can be a simple rule. So, this rule is now generated by the controller. So, earlier this rule was actually inside the routing table. Now this rule is generated by the controller and then the controller actually sends this to the switch and it is installed in the TCAM hardware of the switch.

(Refer Slide Time: 16:39)



Now the switch has this rule; so, once the switch has this ruled in the switch forwards the packet to extreme. Now the rule is already installed in the TCAM hardware of the switch.

(Refer Slide Time: 16:45)



So, that is why for the subsequent packet you do not require to communicate with the controller the communication in the controller is only required for one type. So, you forward it to s 1 and then send it back to h 3 and that is the rule which is being installed in s 1.

And for all the subsequent packet there would be a TCAM it and the cache it. So, whenever there is a cache it you directly forward it to SDN. Now this is this entire h t and architecture and before going to the SDN architecture let me tell you the power of SDN. Now with the help of this dynamic configuration you can actually support lots of new things along with a simple forwarding.

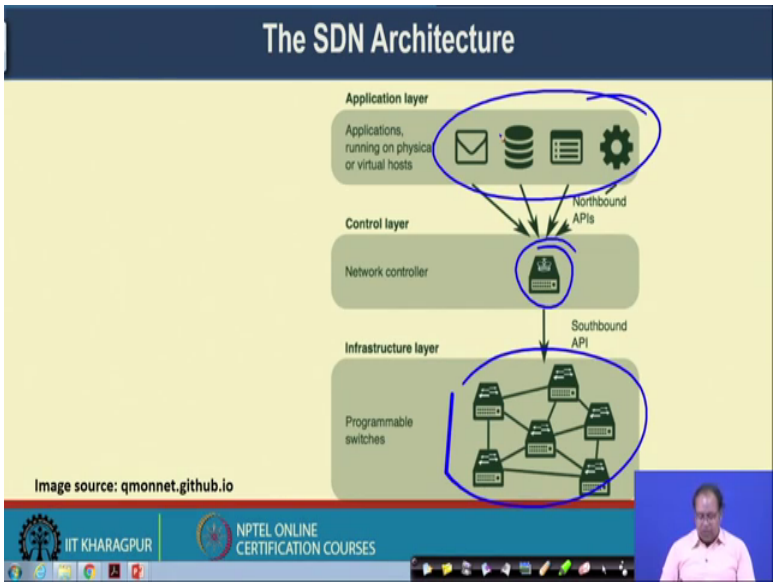
So, now with this match action pair kind of rules you can also implement a firewall. How you implement a firewall? You can implement the firewall is something like this say if your destination IP is 172 dot 16 dot 20 slash 24 then you drop the packet that can be a firewall rule which you can always install inside s 1, inside the switch inside the TCAM hardware of the switch.

So, that way you can design a white class of rules. So, we will discuss in the next class the different open flow supported rules which are there in the open flow standard. And you can actually support a large pool of such rules to implement different kind of network application at the controller. You can implement the firewall, you can implement a NAT, you can implement a forwarding gateway, you can implement a packet gateway. Even you can process because the controller is working at the application level, you can also process at the level of virtual LAN, or at the level of even at the transport layer. You can look into the port and based on the port you can decide what to do.

So, for example, if you just want to ensure that you should not send any packet to port 80. So, you can just write a rule like this say if your port is equal to 80, then you drop the packet well. So, you can also write the rule in this way at the controller side. So, that way you can implement the wide class of network application at the controller and it is not limited only to forwarding under routing behavior.

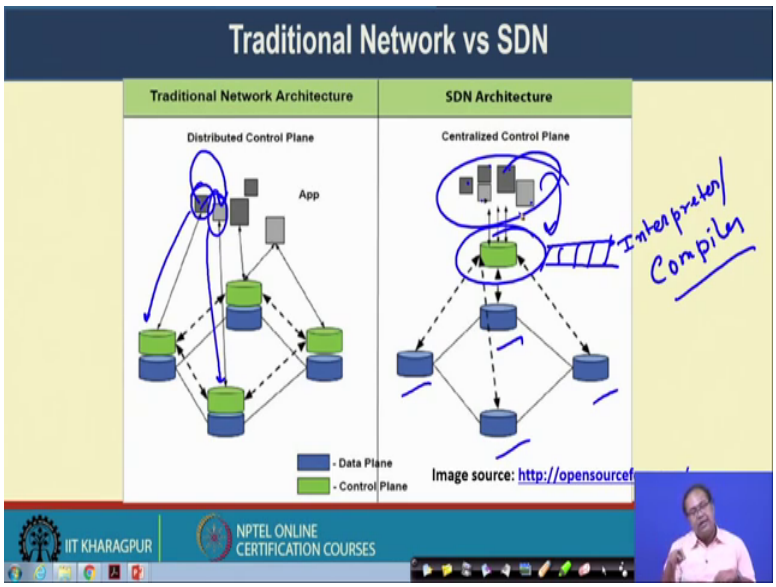
So, ultimately most of the network functionalities you can map it to a forwarding behavior. So, you are deciding how to forward a packet, or you are deciding whether at all to forward the packet or not. So, all these things can be handled by a single controller and that is having a centralized logic because it has a centralized logic. Managing this entire thing is very easy because, nowadays now you do not require this distributed configuration of the control plane of individual routers. Just sitting on a single computer which has a controller software installed, you can implement all these network applications.

(Refer Slide Time: 20:06)



So, this is the broad SDN architecture at the infrastructure layer you have the programmable switches, the different programmable switches which are the dumb switches. But they can be programmed dynamically then you have a network controller at the control layer. And finally, you are running you can run multiple applications on top of this network controller you can implement a firewall, you can implement a custom forwarding engine, you can implement a packet gateway, whatever application you want to implement on top of this controller.

(Refer Slide Time: 20:42)



So, here is the difference between the traditional network and SDN. So, in case of a traditional network you have the control plane and the data plane inside every individual switches. And this control plane they will talk with each other, work in a distributed way and on top of that you have the network applications which are running. And now because these network applications say one network application is interacting with this particular router, another network application is interacting with this router.

So, there can also be consistency problem not the configuration problem, it may happen that this network application is having a conflict with the another network application, and deciding that conflict in a distributed architecture is very difficult. But whenever we are moving to a centralized SDN architecture, or logically centralized architecture. The data planes are distributed well they just implement a forwarding logic, but the control plane is centralized and all the application are actually talking with a single control plane.

Now what you can do this is another power of SDN that you can implement a compiler kind of software here, or an interpreter or a compiler, or a compiler inside this control plane which will generate the rules from individual programs. And then it will also check whether two rules are having a conflicting behavior with each other or not. So, that way you will be able to identify the conflicting rules or you will be able to also manually check whether the rule is actually conforming to the network policy which you want to build inside your network. So, that way this internet work management procedure becomes simplified.

And it provides you a flexible and cost effective architecture to manage a large scale of network. So, that is a brief introduction about software defined networking concept; in the subsequent classes well go to the little details about the software defined networking concept. We look into the open flow standard in detail. So, the open flow standard is a set of protocol or a set of messages which help you to communicate between control controller a centralized controller and a router, or SDN switch in SDN term we do not call it as a router. Because now the routing functionalities are not implemented inside the device we just call it as a SDN switch.

So or sometime it is called open switch. So, we just this open flow controller it designed a set of messages to interact between the controller and a open flow switches, or the SDN

switches. So, you look into the open flow protocol in details as well as we look into certain aspects of SDN in further details. So,

Thank you all for attending this class.