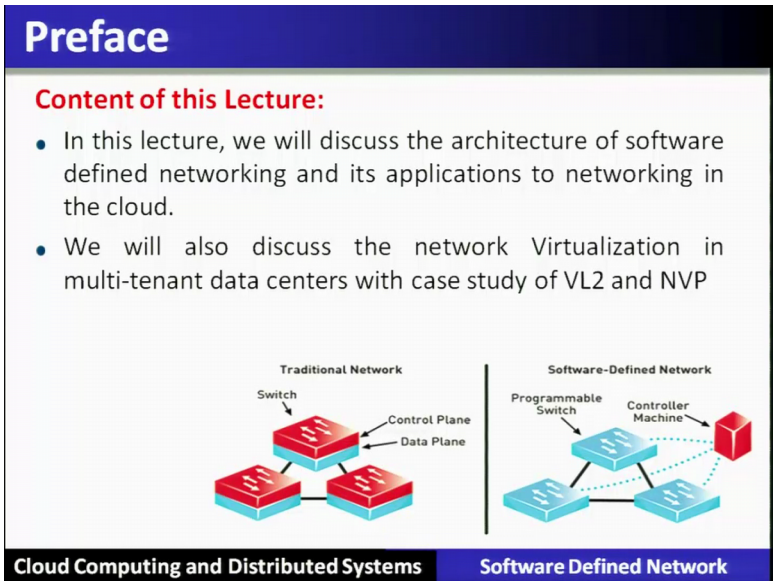
**Cloud Computing and Distributed systems**
**Prof. Rajiv Mishra**
**Department of Computer Science and Engineering**
**Indian Institute of Technology, Patna**

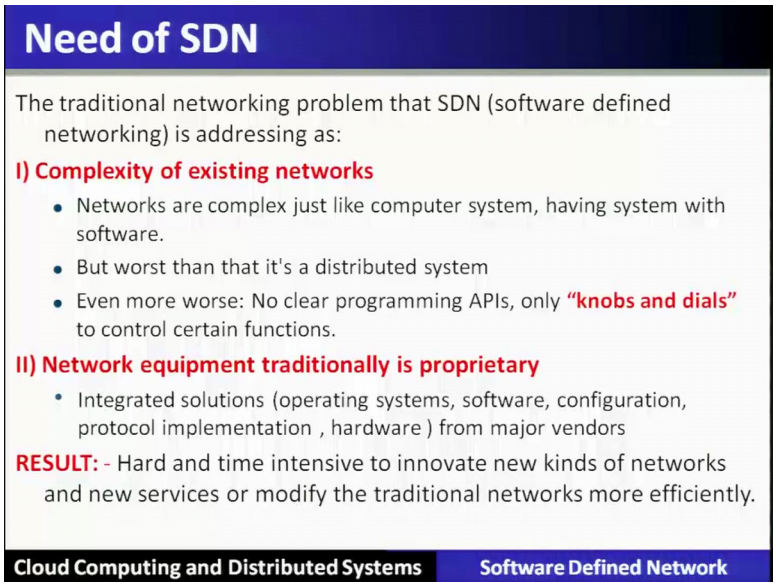
**Lecture – 05**
**Software Defined Network Application to Networking in the Cloud**

(Refer Slide Time: 00:18)




Software Defined Network Application to Networking in the Cloud. Preface, content of this lecture, we will discuss the architecture of software-defined networking, and its application to the networking in the cloud. We will also discuss network virtualization in multi-tenant datacenters with two different case studies of VL2 and NVP.

(Refer Slide Time: 00:42)




The need of software-defined network requires to understand complexity and difficulty of traditional networks. And therefore, it becomes the motivation to understand the software-defined network. So, software-defined network in contrast to the traditional networks will overcome these difficulties in the traditional networking problems, such as complexity of existing networks.

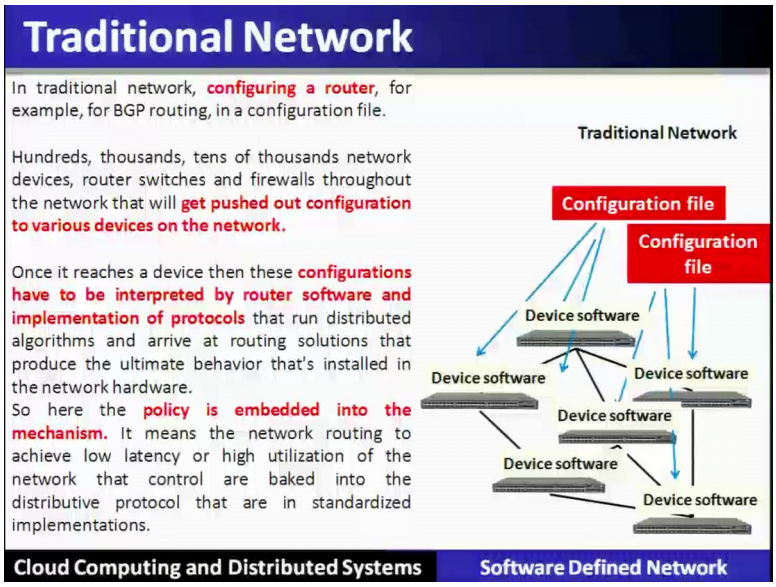
So, the current networks are quite complex; in the sense, it comprises of several 100 networking devices, which are having the embedded software within it, and the supplier has already configured the protocols. So, all the hardware and the software of the network devices are provided by the vendors, they are all proprietary in nature, worst than this it becomes a complete distributed system, wherein the traffic flow and other traffic and other policy decisions over the network regarding the traffic engineering and all these are done in the form of distributed algorithms.

And some aspects are being configured by the vendors. Even worst is that, there is no programming APIs available, only very few control functions are configured in such a network. So, in nut shell, this particular network is very very complex. And if a particular user want the network to be designed regarding the throughput and latency and all these aspects, it is very difficult to program such a so, complexity of the network is known to everyone in the traditional networking.

Now, another aspect in the traditional networks is that network equipments are traditionally proprietary in nature that means, the vendors provides the integrated solution in terms of operating system also is a part of that networking devices, software also is a part of networking devices, the configuration, the protocol implementation, hardware, all are integrated, and that comes from the major vendors. So, therefore, it is very little scope to understand and do the programming of these particular network devices.

Now, why it is required? It is required, because it is the time, when different users like for example, the cloud requires to innovate new kind of rules and methods to configure the network and run the services as per the requirements. But, such a traditional network is very difficult, due to these two problems to modify and give the programming control at the user level.

(Refer Slide Time: 04:28)

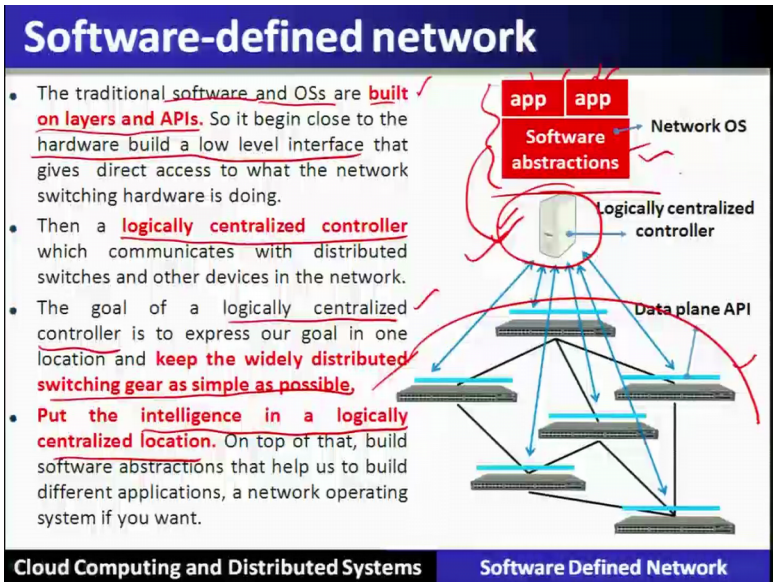



Let us see this difficulty of the traditional network in this scenario here. In the traditional network you can see here, the configuration of a router. For example, if the routers are configured for external routing, that is called BGP routing, and that is done in the configuration files. So, these particular router configured files are then send to or pushed out to the various devices switches. And once it is reaches to these devices, then basically the router software and its protocol will run the distributed algorithm based on that

configuration to arrive into a routing solutions to basically produce the ultimate behavior that is installed in the network hardware.

Therefore, you can see in such traditional networks, the policies and the mechanisms both are embedded into the devices, and the distributed algorithm runs to deal with the traffic flow. Now, if you want to achieve the low latency and high throughput in this scenario, the controls are not given in the form of programming level, but it is already embedded into the distributed protocol, which is basically a very standard implementation, very hard to program and change as far as the requirements of applications are therefore.

(Refer Slide Time: 06:15)




To get this particular flexibility, there comes a rescue in the terms of software-defined networks. So, software-defined networks will deal with these complexities and these problems, so that it can be useful or usable in a different environment that is in the cloud scenario.

Now, let us see: what is the software-defined network, and how it does overcome from the problem of traditional networks. So, analogy is to be taken from the traditional software. So, the traditional software and the operating systems if we see how they are built, they are built on the layers and through the APIs, the programmers can program them that is the software and the operating systems are built in the form of systematically in the form of the layers and APIs are there. Therefore, this particular environment that is

the network environment can we do as per the traditional software and operating system inside this network devices.

If it is then, we are going to see the software-defined network provides this particular separation of policies and mechanisms separated it out in the terms of this hardware of that particular network devices can be accessed in the form of low level interface. So, directly it can be accessed as far as the policies are concerned, they are separated out from this particular logic. Therefore, in software-defined network, there exist a logically centralized controller, which is shown over here, which communicates with the distributed switches and other devices. And all the smart logic is embedded here in the centralized controller. So, as far as the user is concerned it can express all its policies centrally through this logically centralized controller.

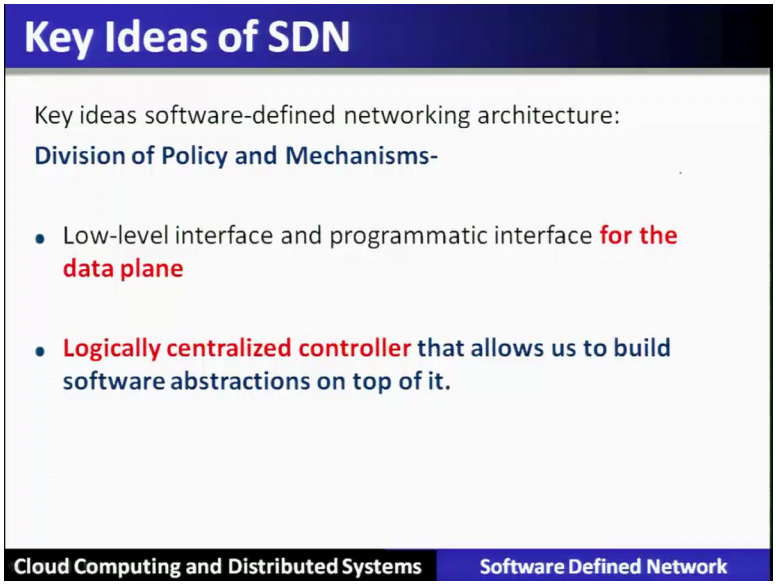
On the other hand, the switches and the other networking devices becomes very simple, because all the decisions are basically the policy decisions and its programming is now taken out from them, and it is basically the pure hardware with the data plane APIs being provided. So, the switching gear is made as simple as possible. So, any vendor can provide its hardware and the logical centralized controller will embed with the software within it. These two separations are there in software-defined network.

So, all intelligence is put in the centralized location. And on top of it is built the software abstractions over which different applications can be made. So, the programming of this logically centralized controller is managed just like the softwares, which we have seen that it is in the form of layers and APIs, which can be programmed as per the need of the organization to run the networks.

So, you can see here, there is the logically centralized controller, and these particular devices are simply the hardware, which provides the data plane APIs, and it gets the instruction from logically centralized controller and work accordingly. So, now the control is all the policies are with the logically centralized controller, which can be programmed using these two layers that is in the form of software abstractions and different applications can run on top of it, which can use this network the way they want to. This software abstraction can be the network operating system and different applications can be programmed as per the requirement of the network. So, this is the

brief overview of a software-defined network. Let us see how this software-defined network will be useful for cloud networking in this discussion.

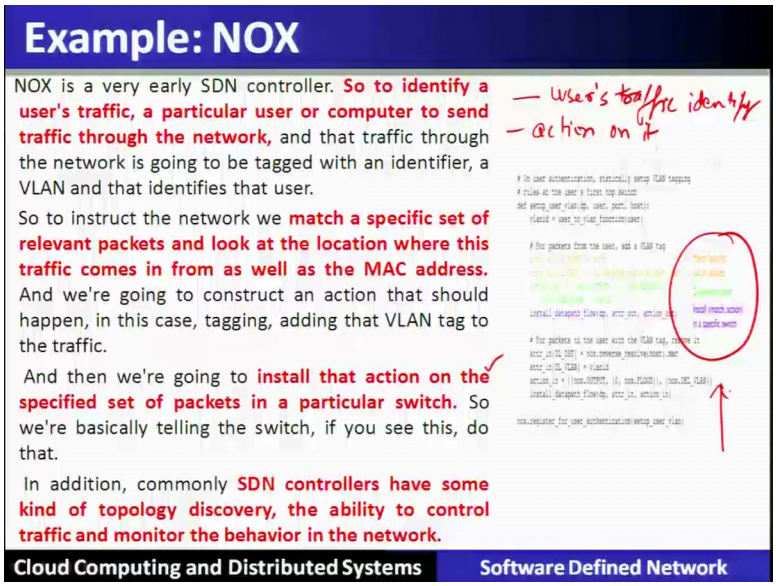
(Refer Slide Time: 11:17)




So, again let us summarize the same thing, key idea of a software-defined networking architecture is to divide the policy and mechanism. So, that the low level interface and programmatic interface is provided for the data plane, and the policies are controlled in the form of logically centralized controller that allows to build software abstractions and users can program it.

Now, in the diagram, we have shown only one logically centralized controller, it is not a centralized controller; it is a logically centralized controller. In the sense, there may be more than one controller or the controllers are also run in a form of a distributed system, they may communicate with each other, they may be synchronizing their clocks with each other. And together they may solve or may communicate with each other to enforce the mechanisms and the policies given or programmed, so that all the network devices can work accordingly that means, all the devices can be programmed using the data plane APIs.

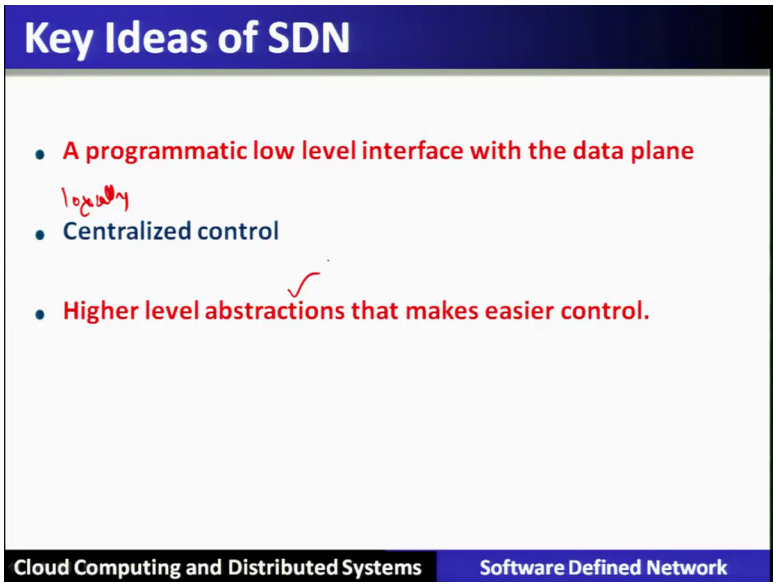
(Refer Slide Time: 12:35)




Let us see an example of the very earlier system of SDN controller, which is called a NOX. The example, which we will give is that how the NOX controller will identify a particular users traffic, and then enforce some action on it, in traditional network that is not possible to identify a particular users traffic and take action on it. Let us see how software-defined network in controller that is NOX controller does that. So, to identify a users traffic, a particular that is a particular user or the computer to send the traffic through the network, then that particular traffic is tag with an identifier in a virtual LAN, that will identify the users.

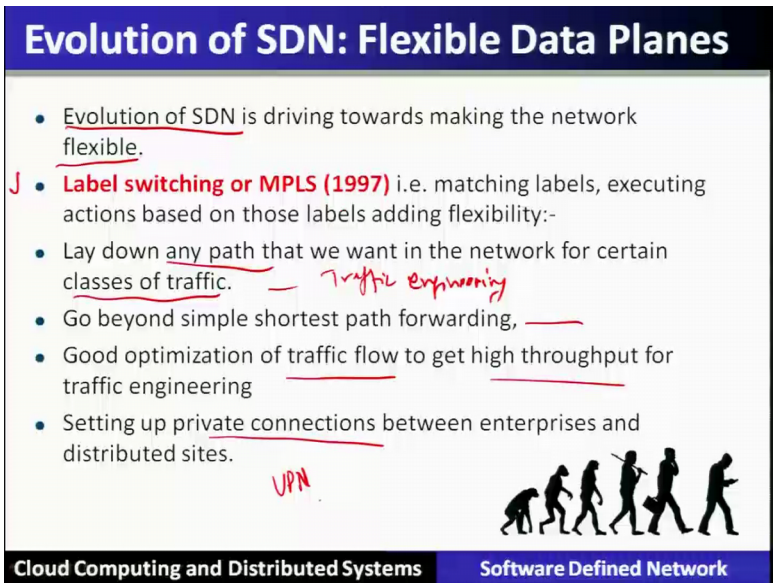
Now, the other aspect is to instruct the network that once this particular traffic comes out from that particular user, so that means, there is a match on a particular set of packets originated from that tagged user based on the MAC addresses and some other identifications. Then what action has to be taken up. So, the actions are also installed, which will specify what to do with those particular packet, when it arrives a particular switch, whether it is to be dropped or it is to be forwarded all these aspects can be programmed here in this particular scenario. Therefore, using SDN controller, many things can be done for example, monitoring the behavior of the entire network and ability to control some traffic are also topology discovery can be done through this particular way of software-defined networks.

(Refer Slide Time: 14:56)




So, again the key idea is that programmatic low level interface is being provided with the data plane, so that the hardware switches can be used at as the APIs, so any vendor can pick in providing their hardwares in the form of switches, routers and so on. The entire control is done in a centralized manner that is the logically centralized controller existed. And to program this, there is a high level abstractions, which are provided just like a software, which can be programmed using the simple languages like python. So, the entire network can be programmed as per the requirement.

(Refer Slide Time: 15:47)

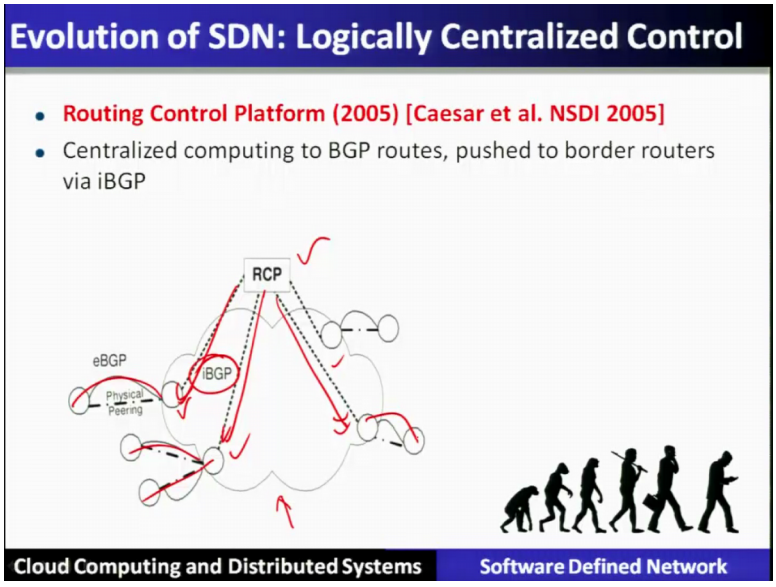


Let us see the evolution of software-defined network, which will provide the flexible data planes. So, the evolution of software-defined network, as we have just seen is to provide the flexibility in the network operations, so that different users can program as per their requirement the entire network, and will overcome from that proprietary in nature. So, the evolution that means, let us trace through the evolutions, which has reached to this particular stage that is software-defined network.

The first one in terms of the idea was called MPLS multiprotocol label switching in 1997, what it does it, matches with the labels, the traffic, which can be used for example, to setup a virtual private network, and then perform various actions on such particular traffic. So, this way one can lay down any path in the network to classify that particular certain traffic class, and do the traffic engineering on it.

So, MPLS provides these particular features. So, it is going beyond the shortest path algorithm, because here lot of traffic engineering can be performed. Similarly, lot of optimizations on a network flow can be done, so that there is a possibility of achieving the throughput. And also this is used to setup virtual private network connections across the different enterprises, and this is all possible using MPLS. So, MPLS has started giving this kind of provisions, and this is by start point of evolving the software-defined network.

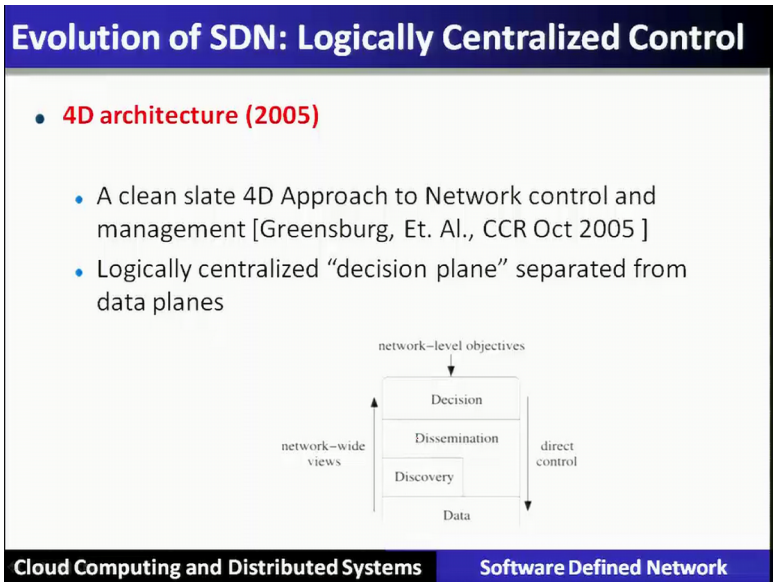
(Refer Slide Time: 18:10)



The next one in this evolution is called routing control platform in 2005. What it provides is that, within an organization also there are many number of border gateway protocols, there are various routers installed with BGP protocol to interact with the outside world. Normally in a routing control platform, that means, at the centralized control, the policies are being pushed to these border routers via iBGP. So, this iBGP intern will propagate to the other BGPs, which are installed externally or which are connected to the other BGP.

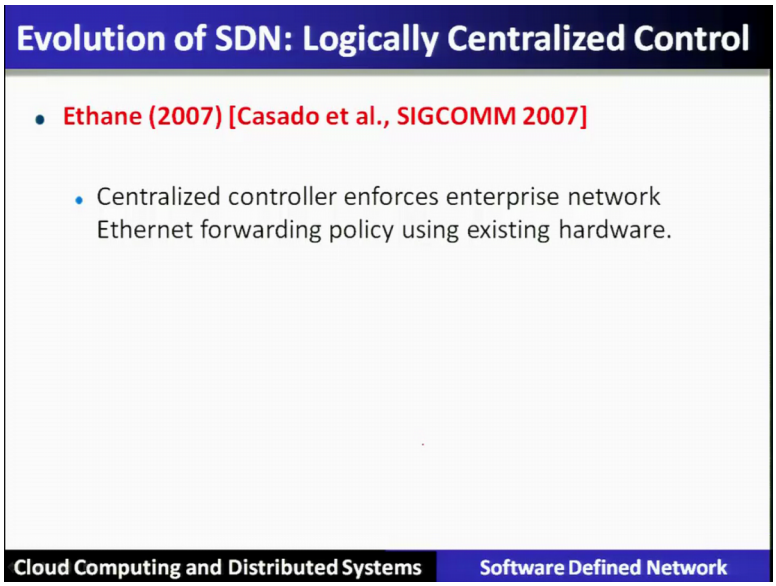
So, here we can see that the explorer traffic, that means, when the organization is communicating its messages with the other networks, that it can be programmed through the central control, through the centralized computing for computing the BGP routes and following the routing information. So, this is also the aspect, which is required here in the evolution of software-defined network.

(Refer Slide Time: 19:56)



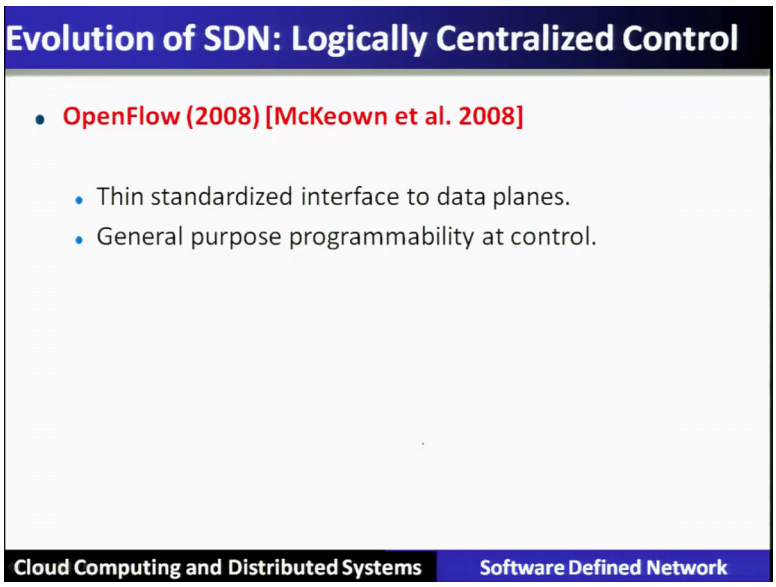

Another, architecture is 4D architecture of 2005. Now, here it gives a approach to the network control and management. So, it also defines logically centralized decision plane separated from data plane. These ideas are also there in software-defined network.

(Refer Slide Time: 20:16)



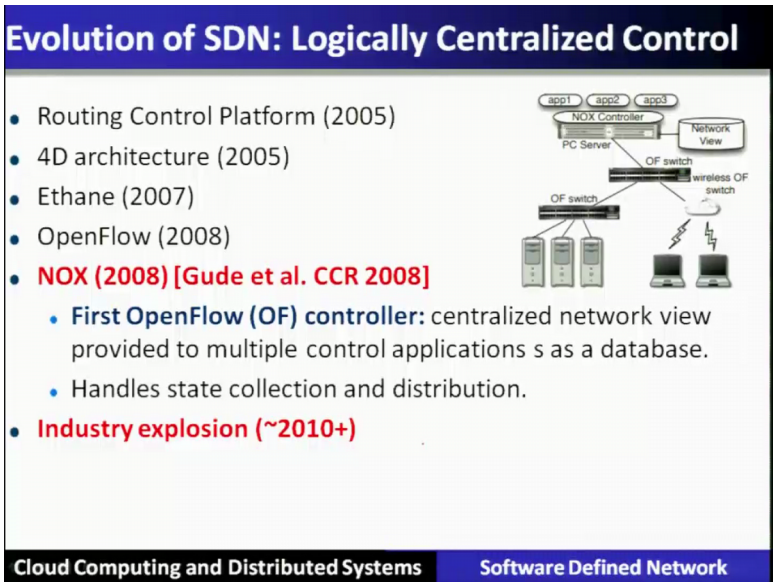

Ethane is just a precaution to the SDN; ethane was given in 2007. It is a centralized controller, which enforces the enterprise network Ethernet forwarding policy using existing hardware. So, again it is separating out policies and mechanisms.

(Refer Slide Time: 20:36)



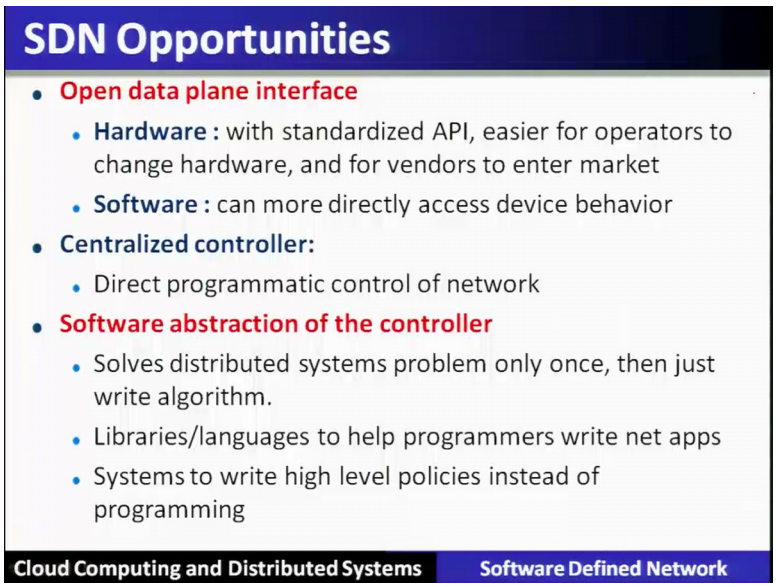

Now, comes after that is open flow in 2008. This is a thin standardized interface to the data planes; and it also provides a general programmability at the control level.

(Refer Slide Time: 20:49)




Now, the NOX we have already seen uses the first open flow controller, which is having the centralized view, which is provided to the multiple control applications just like a database it handles. So, there is a central controller, which will have the complete view or it can be programmed as a database does. And what it does it collects by state of the entire network, stores in the database, do the analysis, and provides control back to those particular devices. And after that almost every networking industry has adopted the software-defined network in one form or the other.

(Refer Slide Time: 21:43)

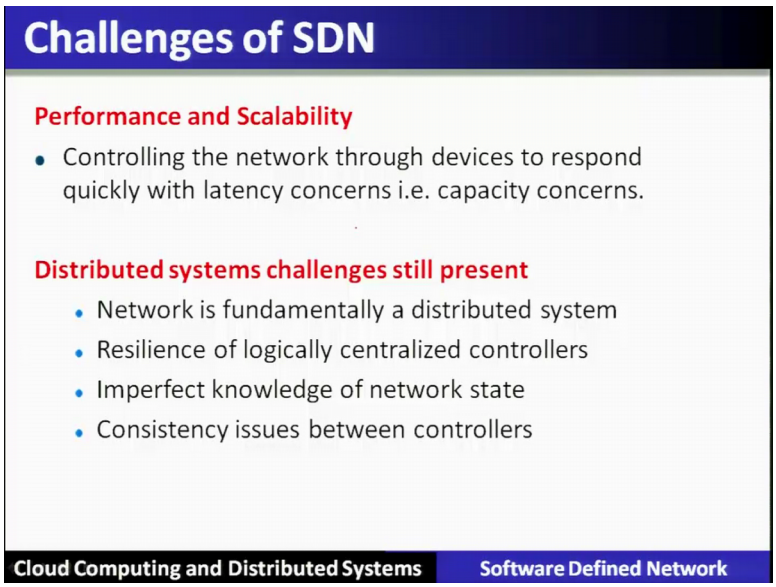


What are the options or opportunities, which software-defined networking is giving us to the networking of the organizations and the cloud as well as. Open data plane interface provides the separation of hardware and software. In the hardware, with standardized APIs, if it is provided, then basically it is easier for the operators to change the hardware, and it is not proprietary hardware, and different vendors can enter into the hardware market. As far the software is concerned software can be programmed to directly control the behavior of such devices. For that, the policies are given from the centralized controller, which also is having a direct programmatic control of the network.

Software abstractions are also supported or provided for the input to the controller. This way, it is all (Refer Time: 23:01) distributed problem only once and then those policies are now generated in the form of distributed algorithm and being pushed to those hardwares, which will use its software to run the mechanisms. For that controls, the libraries, and languages, are also there, and it can be programmed to write down the network operations or applications.

So, here a very high level policies can be implemented in the form of the network controller, and the organizations through the simple programming language, so this way the SDN has given lot of opportunities to program and control the networks the way is required in the applications or in the in the organization.

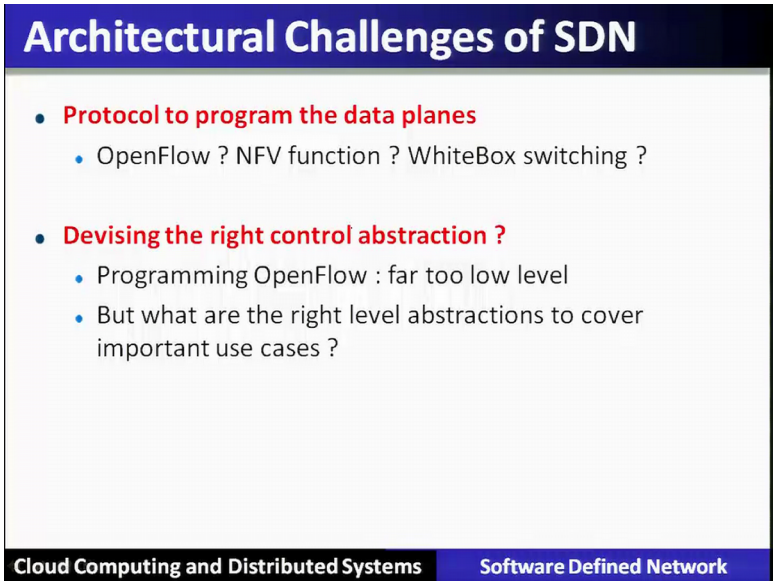
(Refer Slide Time: 24:06)



Now, another challenge of software-defined network is the performance and scalability. So, the controlling such a network through the devices to respond quickly with the latency concerned, and also with the capacity concerned deals are related with the performance and scalability.

So, network is a distributed system. And resilience of logically centralized controller is very much required. So, imperfect knowledge of network system states due to the distributed system is already there, and lot of consistency related issues between the controllers are also there. For example, it is not a centralized one controller; it is a logically centralized controllers several controllers are there. So, all these challenges are still present in the software-defined network.

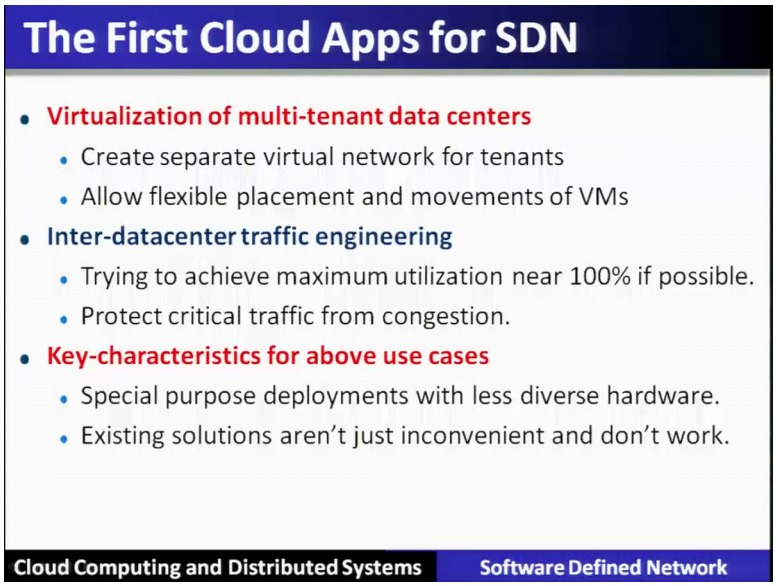
(Refer Slide Time: 25:14)




Let us see some more architectural challenges of software-defined network, how to program the data planes. So, that it can run different protocols, so different mechanisms are there, whether open flow should be used or network function virtualization, NFV can be used or white box switching is there. There are various different alternatives.

And this particular space is open for the research between (Refer Time: 25:47) and the industry. That is why, the devising the ride control abstraction in the form of whether the open flow is already very low level way of solving this particular problem. Instead of that, what kind of abstraction to cover the most important use cases is going to be important development for the future applications?

(Refer Slide Time: 26:20)



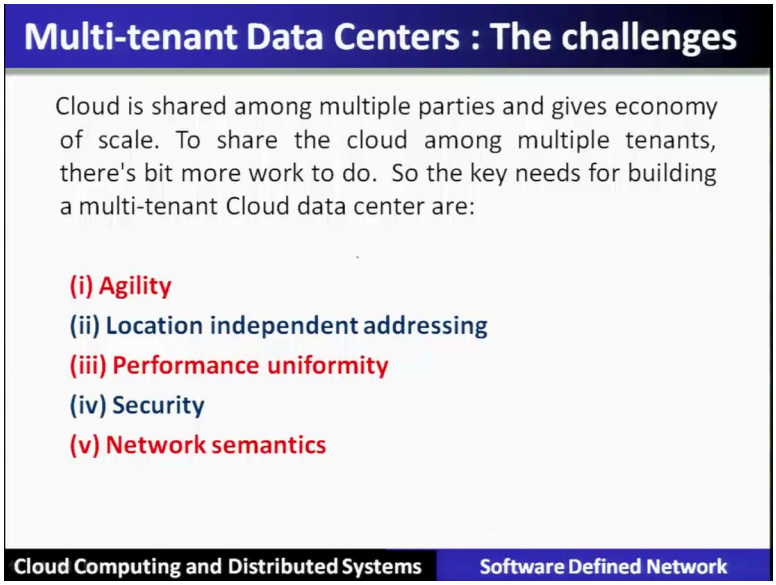

First cloud application for software-defined network. We will see here, the virtualization of multi-tenant datacenters, which is an application of software-defined network into the cloud system. Here we will see that how to create separate virtual functions for tenants, and allow the flexible placement and movements of virtual machines.So, to understand this, in a cloud datacenters, there are many tenants, which are running their VMs. And there is a requirement of the isolation separation among their, the traffic and also the different VMs not shared, but it has to be secured.

So, the virtualization of multi-tenant data center is going to be an important issue. And that is how using software-defined network this particular problem is going to be addressed, that we will see. We will also see the how inter data center traffic engineering can be used. So, that there should be a maximum utilization of nearly 100 percent to achieve the profit of running the data center or the cloud services for the public or maybe for the internal purpose. So, in that scenario, we will see how to protect the traffic, how to monitor the traffic and how to protect it from reaching into a state, which is called a congestion.

We will also see some of the key characteristics for different use cases. For example, this special purpose deployment with a less diverse hardware sometimes is required. Similarly, another use cases existing solutions are not just inconvenient and do not work.

So, how special use cases are going to be addressed in this particular way of software defined networks.

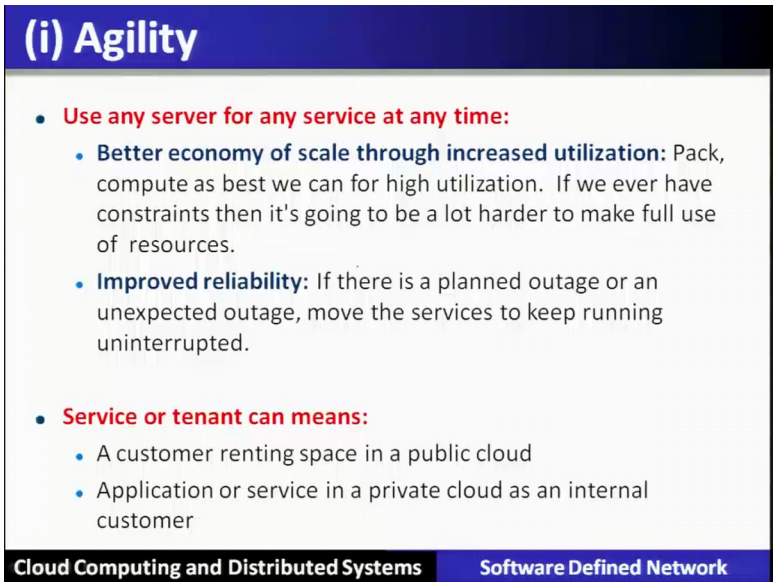
(Refer Slide Time: 28:40)




So, multi-tenant data center, let us see the challenges. So, cloud is shared among multiple parties to give the economy of scale that means, the more are the parties, which are sharing the data center, the profit runs according to that scale, that is called economy of scale in the cloud. Now, to share the cloud among the different multiple tenants, lot of work has to be done start (Refer Time: 29:11) forward.

So, the key needs for building multi-tenant cloud data center are, it has to ensure or provide these five different requirements or we can say that the multi-tenant data center has five different challenges; let us see one by one, agility, location independent addressing, performance uniformity, security and network semantics.

(Refer Slide Time: 29:43)

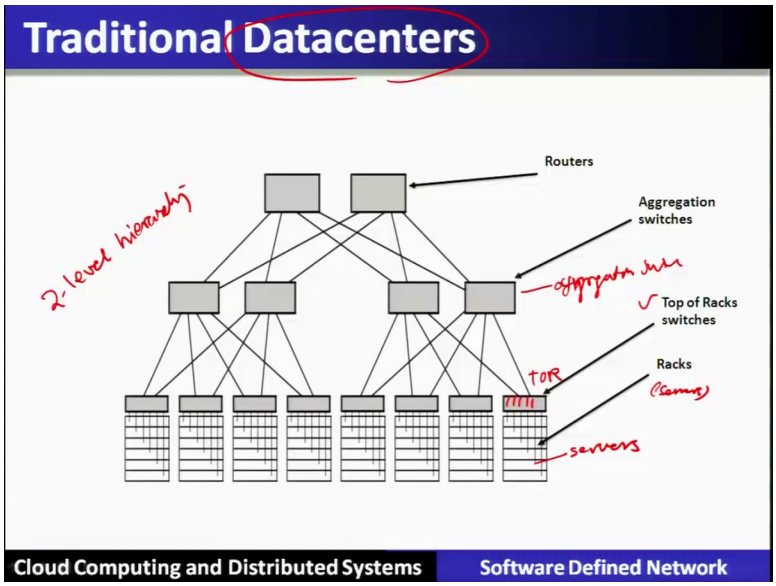



Agility means, that use any server for any service at any point of time. If there is a constraint, obviously, economy of scale will not be achieved, and hence realization of multi-tenant cloud system will not be possible. So, agility is first and fore most important requirement for multi-tenant cloud data center. So, to provide the use of any server for any service at any point of time, means that it will provide the better economy of a scale through increased utilization means to pack compute as best as we can for higher utilization. If you ever have the constraints, then it is going to be a lot harder to make the full use of the resources.

The other thing, the agility will support is improved reliability, for example, if there is a tenant, which experiences the outage which is a planned outage let us say or maybe sometimes an unexpected outage, there in order to run its services without interruption, it can move out its services to some other data center. That is only possible due to the agility, that is use of any server for any service at any point of time, it can move to at any place, which is which can continue to give that particular service, that is called agility.

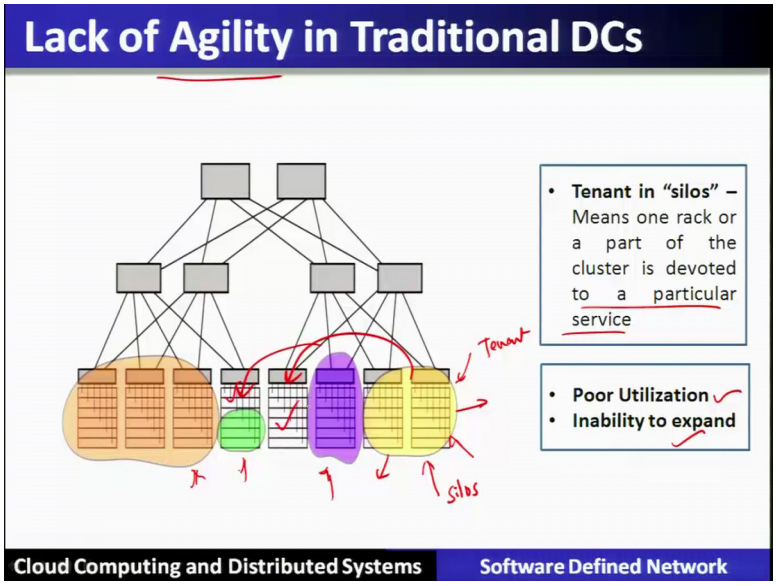
So, the service or the tenant can mean, the customer renting a space in a public cloud, and customer or the service in a private cloud as an internal customer. Either, so either it is a public cloud or internal or a private cloud, multi tenancy can be supported in the form of so, the first requirement is called agility.

(Refer Slide Time: 31:50)




Let us see, what are the issues. This is the picture of traditional datacenter, which comprises of the racks full of the servers, every rack has top of the rack switch. And different racks are connected through an aggregation switch, and they are being connected over the routers. So, it is a hierarchy of 2-level hierarchy, which you can see in any the traditional data center. So, hundreds and thousands of servers are deployed in a particular datacenter, which are connected in this network form.

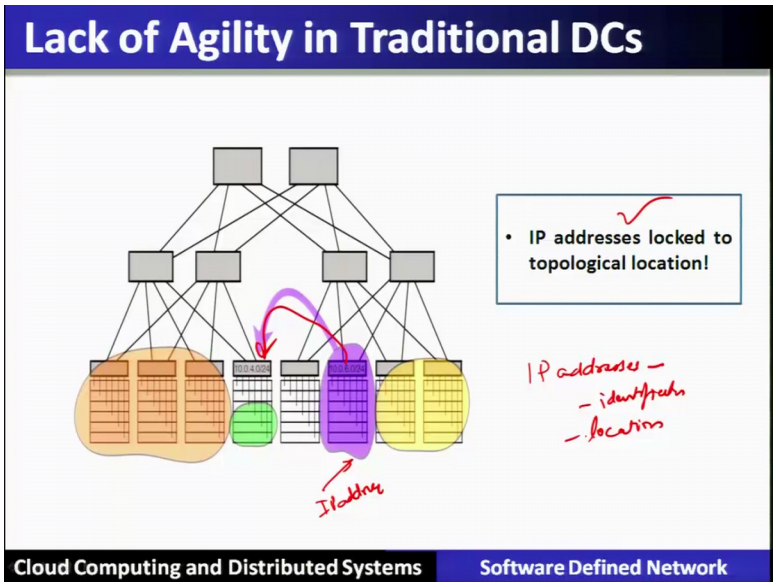
(Refer Slide Time: 33:03)



Now, in this particular traditional datacenter, whether it lacks the agility, let us see why. So, tenants in the datacenter is just like the silos that means, a particular tenant is using let us say these two racks, which are shown here in different colors, so that means, these servers are devoted to that particular tenant service.

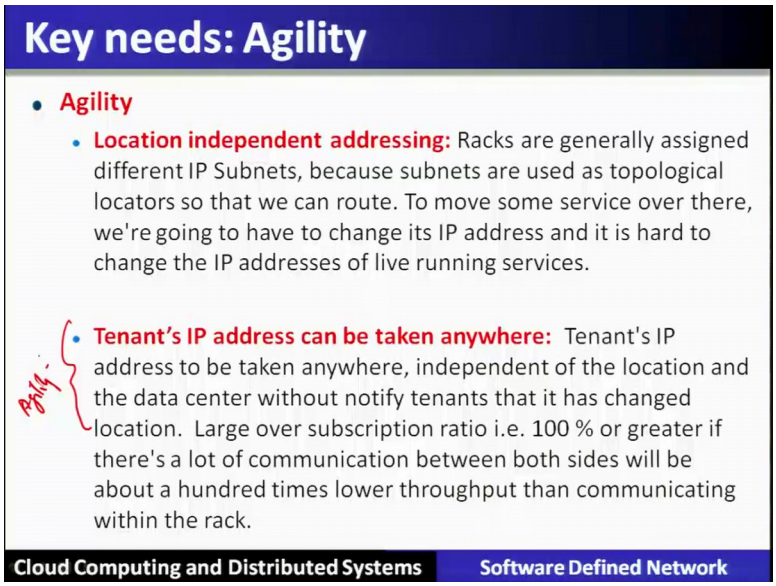
Now, if let us say this particular tenant want to expand its services, which is already full, so that means, which our rack having an empty servers, so how it can be further moved to use that empty scenario, so that is not possible in the traditional datacenters. Why, because it lacks the agility. So, this will lead to a poor utilization, and this inability to expand. Similarly, there are some unused servers, due to this there is a requirement of expansion, expansion is not possible, and also it results into a poor utilization. These are all possible these are all possible, due to the reason that it lacks the agility.

(Refer Slide Time: 34:43)




Why this particular problem is there, why, because every rack and its servers, they are locked they are tied with this IP address. So, moving it to some other rack that means, the applications will breakaway with this particular IP addresses and will disrupt the application in its continuation. So, therefore, this IP addresses are locked to the topological locations. So, IP addresses are over used, that means, it is not only went for the identification, but it also identifying the location; so, it is overly used and not supporting the agility.

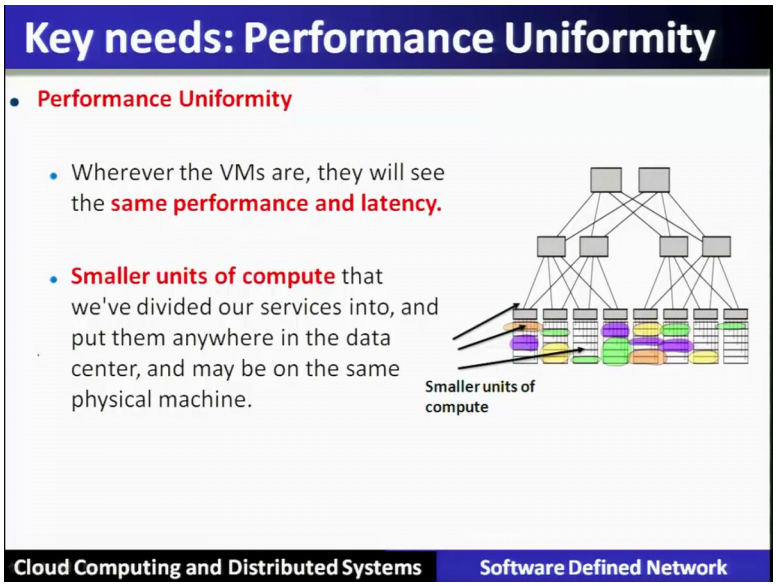
(Refer Slide Time: 35:50)




So, here let us see the details of requirements of the agility that is the location independent addressing, so that means, the addressing should be location independent, that is the racks are generally assigned different IP subnets, because subnets are used as topological locators, so that we can route. To move some service over there, we are going to change the IP address and it is hard to change the IP address of a live running service. So, there will be a possibility of disruption, that is not possible.

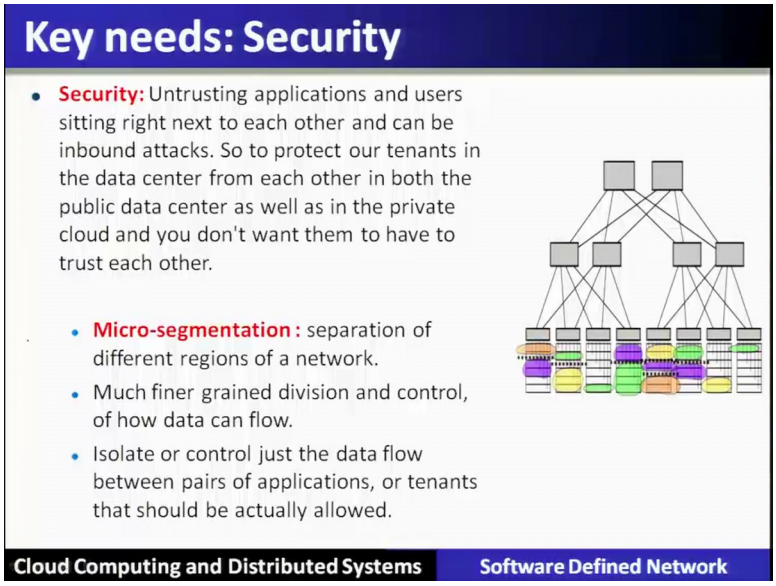
Now, tenant's IP addresses can be taken anywhere, if this is detached with the location. So, tenant's IP address can be taken to anywhere, independent of the location and the data center without notifying the tenants that has changed the location. This is the aspect of the agility, which is to be ensured.

(Refer Slide Time: 37:11)



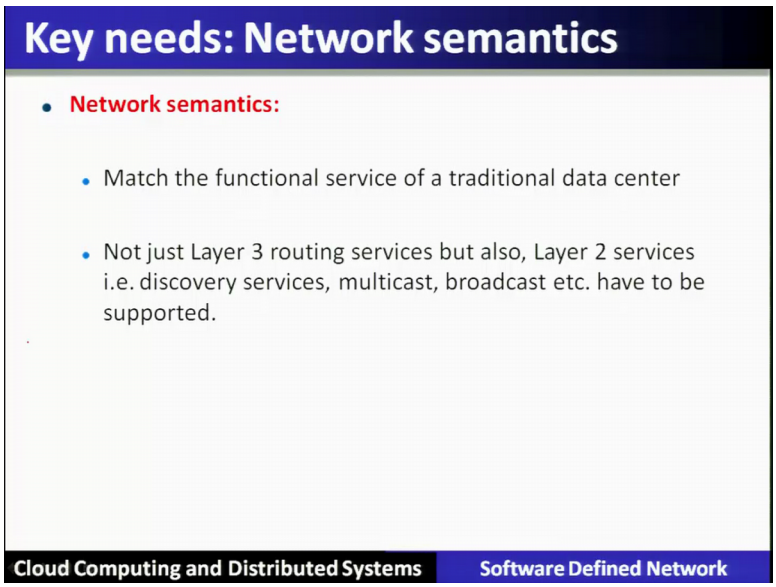

Now, another key requirement is called performance uniformity. We say that wherever the VMs are, they will see the same performance and latencies because, when agility is being provided, so they will be the traffic order data will be segregated. So, all the VMs, all the servers, will be properly used, and they will experience in the same set of same performance and latency. So, smaller units of compute that we divided into the services into, and put them anywhere in the data center, may be on the same physical machine.

(Refer Slide Time: 37:55)



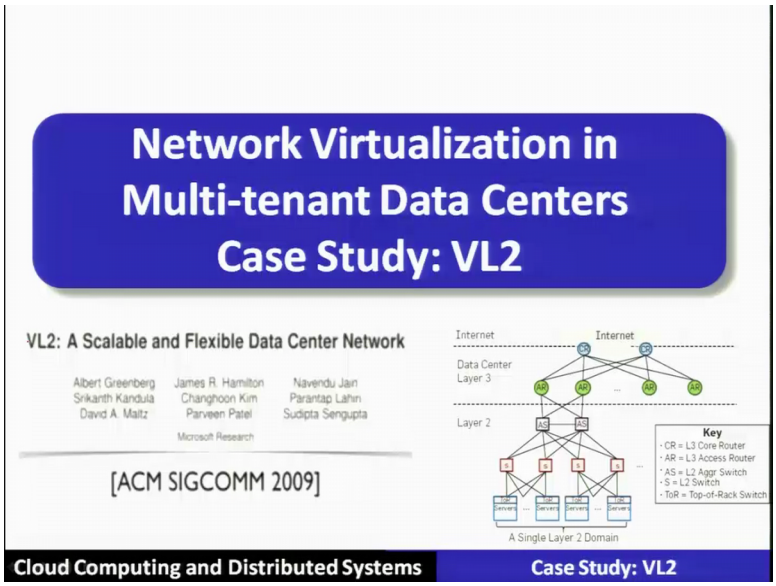
Now, third requirement is called a security, that is untrusted applications and users sitting next to each other can face the inbound attacks. So, to protect the different tenants into the data center from each other, you require the security is to be enforced or implemented. So, there are various techniques, for example, micro segmentation. The separation of different regions of the network; and much finer grained division and control of how the data can flow and isolate the control just the data flow between the pairs of application, or the tenants. So, there are different ways are there, but the key accept here is to ensure the security.

(Refer Slide Time: 38:56)



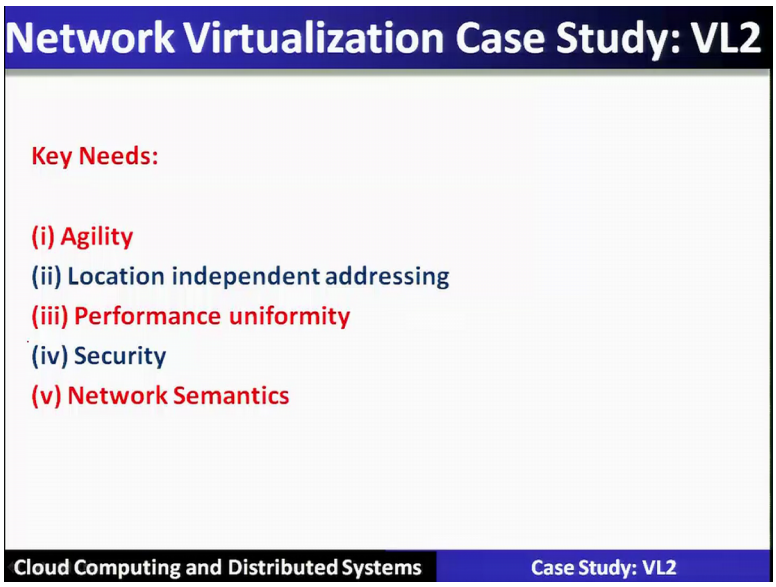

Finally, the last requirement of multi-tenant datacenter virtualization is called network semantics. So, it will match the functional service of the traditional datacenter, that is the traditional data services which are supported either in the layer 2 or layer 3 services in the form of discovery services, multicast, broadcast also has to be supported.

(Refer Slide Time: 39:30)




So, how this particular four five aspect of multi-tenant datacenters are to be virtualized that we will see in two different case studies, one is in the paper, which is described as VL2 method, that is the scalable and flexible data center design.

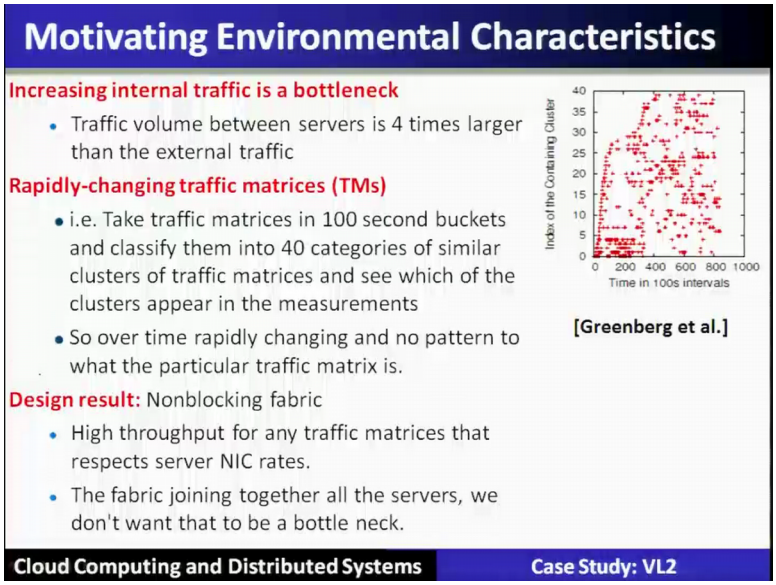
(Refer Slide Time: 39:53)




Let us see how the VL2 provides the multi tenancy virtualization for data center. [FL] let us take a break. Network virtualization in VL2; Again, we will see how the five different key needs for multi tenancy datacenter virtualization is being architect or being designed in VL2, that is we will see how the agility is supported, how the location independent

addressing is provided, how the performance uniformity is guaranteed, and how the security is ensured, how the network semantics is being allowed.

(Refer Slide Time: 40:49)



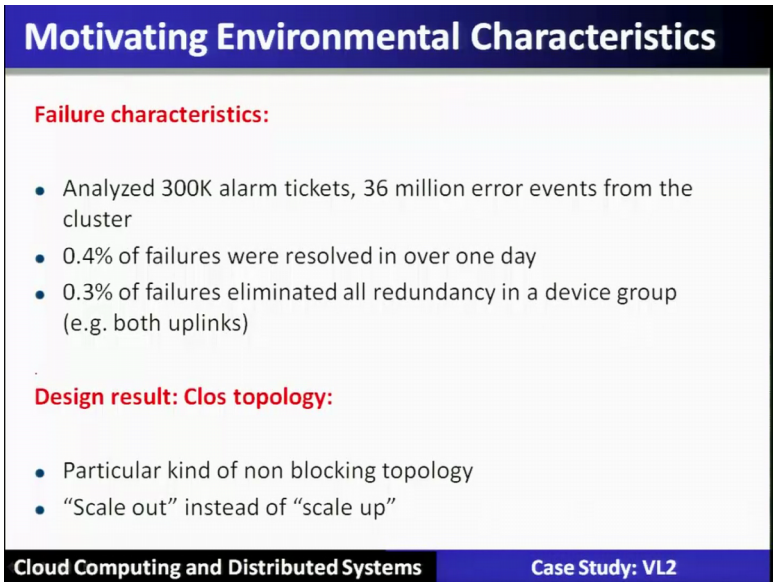

Before we go ahead into the details of the mechanisms, we will see: what was the motivating environment that led to the development of VL2 mechanisms. They have analyze the internal traffic, and found that increasing internal traffic is a bottleneck. Why, because the traffic internal traffic within the data center between the servers is four times larger than the external traffic that is the traffic outside the data center. So, this particular internal traffic, which is becoming a bottleneck, has to be at rest, let us see that.

Now, they have also analyzed this internal traffic through the traffic matrices, and they found that this particular traffic matrices is rapidly changing, there is no fix pattern. How they have analyzed is they have taken the traffic matrices in 100 seconds bucket and classify them into 40 different categories of clusters of traffic matrices, and see which of the clusters appear in the measurements. If you see the diagram, it is all is scattered that means, there is no way a particular cluster is appearing again. So, over the time rapidly changing and no pattern to what the particular traffic matrix is, so that means, you cannot fix up a particular traffic matrix, it is basically rapidly changing even in the internal traffic, which is a bottleneck into a data center.

So, therefore, the approach for the design should be that the internet or a data center network should be non-blocking fabric. And it should basically be a high throughput for

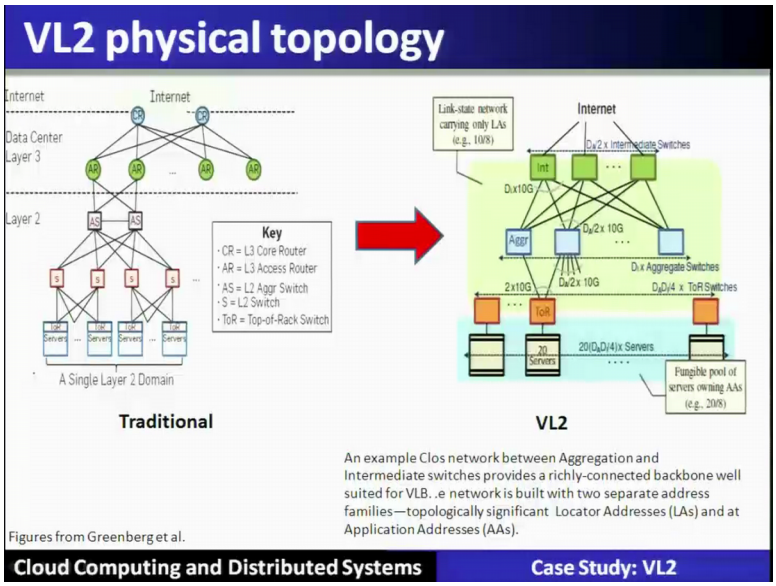
any traffic matrix irrespective of the server NIC rates. The fabric joining together all the servers, we do not want that it to be bottleneck in any case, so that brings up the motivating point that how the non-blocking fabric, which will provide the high throughput for any traffic matrices being supported as per the design is concerned.

(Refer Slide Time: 43:29)



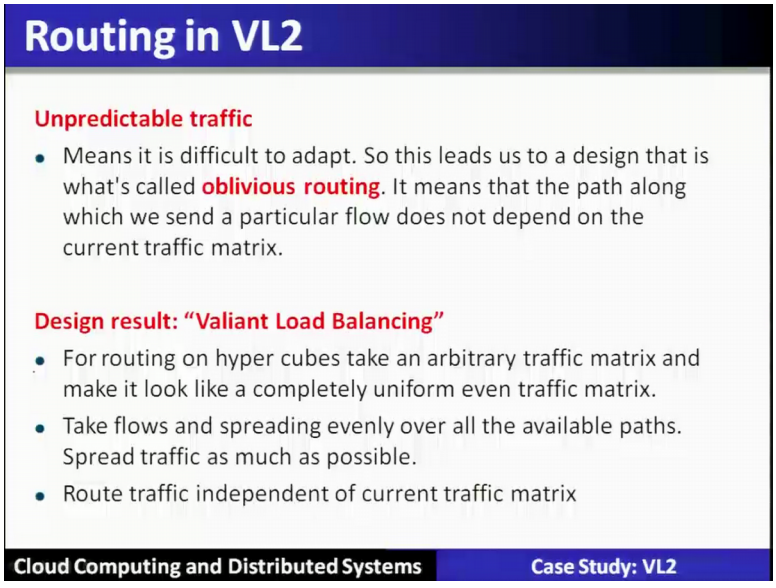

Now, another aspect, which is being seen here is about the failure characteristics. There are several error events occurring that is 0.4 percent of those failures can be resolved in a particular day, 0.3 percent of the failures eliminated all the redundancy in a device groups. So, as far as these failures and their handling, this is specific characteristics has shown that the design requirement should be like clos topology that is a particular kind of non-blocking topology, which will be scaled out instead of scaling up, so that means, large number of devices should be provided in such a topology.

(Refer Slide Time: 44:24)




Let us see the VL2 physical topology. Now, in the VL2 physical topology, we will see that different devices, which are connected that means, using different kind of devices are there, hierarchically networking the entire data center that is the racks and the servers.

(Refer Slide Time: 44:46)

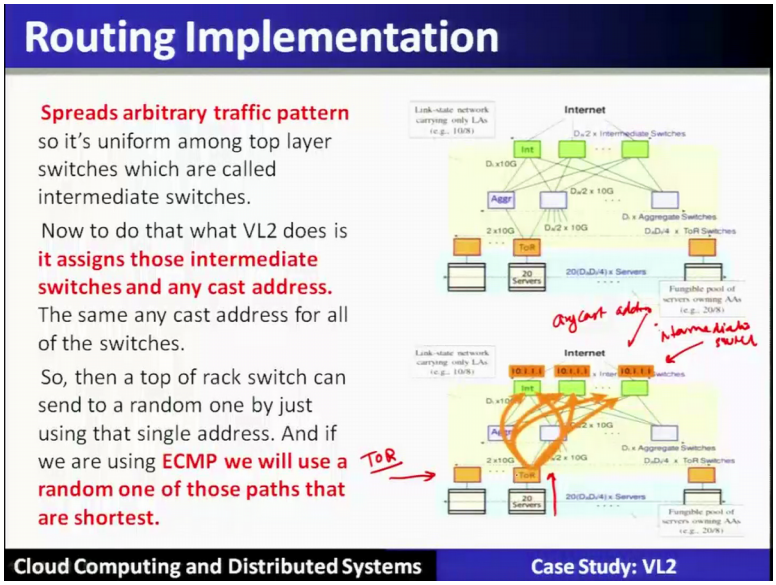



Now, let us see in this particular scenario, how the routing is supported here in VL2. Since, you know that we have already seen is a unpredictable traffic matrix, so best thing is to use the oblivious routing. Oblivious routing will basically not deal with the

particular traffic flow, rather than it is independent to the traffic flow and that means, the path along which we send the particular flow does not depend on the current matrix. So, oblivious routing is basically supported in this scenario.

The other designed result is a valiant load balancing. So, valiant load balancing also does not takes into an account of a current traffic matrix. So, for an arbitrary traffic matrix, the routing on a hypercube will give the valiant load balancing. So, take the flows and spreading evenly over all the available paths, so that becomes a multipath kind of routing a valiant load balancing will be done, so that means, is spreading the traffic as much as possible in the valiant load balancing. And route the traffic independent of the current traffic matrix.

(Refer Slide Time: 46:05)

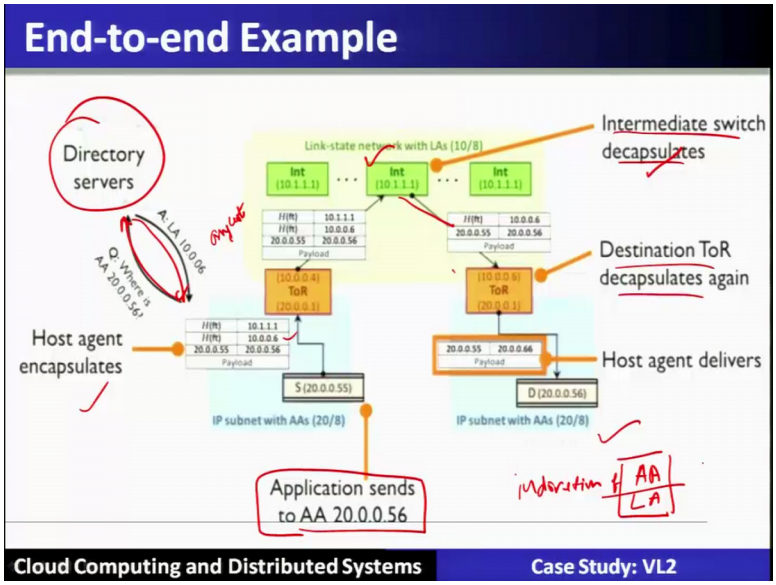



So, it is so after spreading arbitrary the traffic pattern, so it is a uniform among all the top layers, which is which are called intermediate switches. Now, to do that what VL2 does is that it assigns those intermediate switches with any cast address. So, then the top of the rack switch can send to a random one by just using a single address. We are using ECMP; we will use the random one of those paths that are shortest. Let us understand these concepts.

Now, here they are called intermediate switch. And any cast address is being used here, when a packet reaches up to this point. As far as the top of the rack switch, when a

packet originates from the server goes to the top of the rack switch, then top of the rack switch will change this particular address.

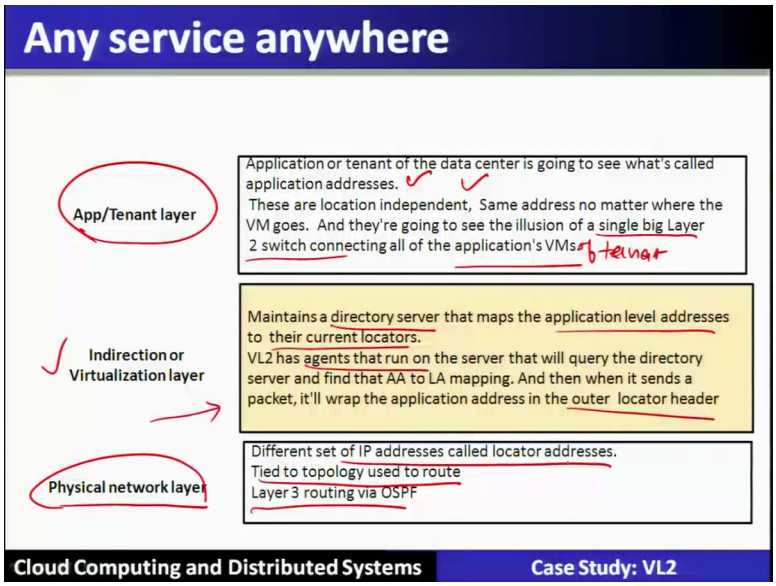
(Refer Slide Time: 47:18)




Let us take this end-to-end example. So, whenever a packet comes, then it will encapsulate into an application address like this. So, it will be wrapped or encapsulated with that address. To know this address, the top of the rack switch will basically consult a directory server, and the directory server will give the application address, which will be used up. So, this particular application address will be the any cast address, and will reach to the intermediate switch like this. Now, this intermediate switch will decapsulate the application or any cast address, and it will send to the destination top of the rack switch here in this case, which decapsulates the header of a top of the rack switch, and finally, it will be delivered to that application, which they want to interconnect.

So, using the directory service, this indirection of the application address with the logical address is being separated out. Therefore, the IP address is now is being used for internal communication as you have seen when it reaches to the top of the rack switch within that IP addresses will be used; But, across the communication, application addresses are used and that is the dissolved using the directory service. So, therefore, this is going to be supporting the agility that means, it is not tied up with a particular rack address or IP address, so that the VMs can move from one server to other server or one rack to other rack in the data center.

(Refer Slide Time: 49:35)




So, this particular aspect is being supported by the virtualization layer. So, therefore, there are three different layers. Now, earlier we have seen there will be a physical layer, which will be nothing but the IP addresses, which are also called the locator addresses, and they are tied up on the topology used, and it also follows a particular routing scheme that is called OSPF at the physical network layer.

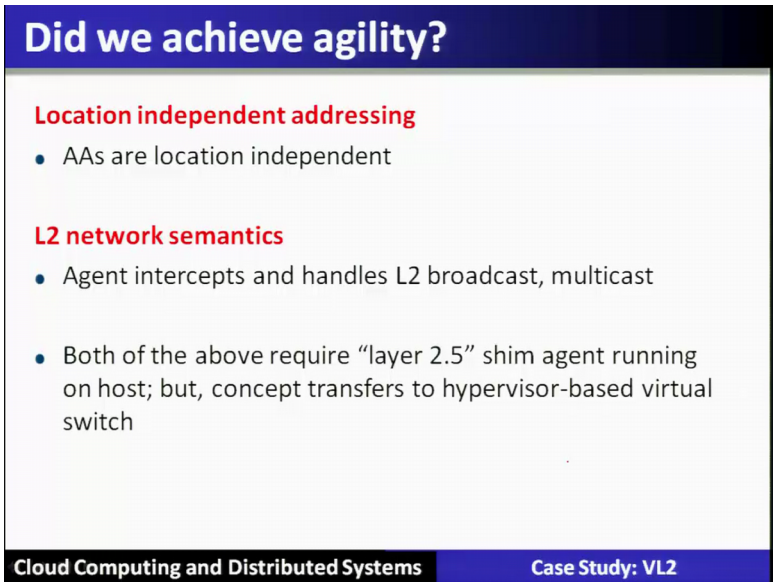
Similarly, at the application layer or the tenant layer, so the application or tenant of the data center is going to see what is called the application addresses. So, that tenants, they can communicate through the application addresses, which is quite different than the IP addresses. So, these are location independent addresses. So, same addresses no matter, where the VM goes, and they are going to see the illusion of a single big layer to switch connecting all the application VMs. So, now therefore, all the tenants can communicate with each other, and they will not cross across to the other tenant traffic. So, therefore, the illusion will be that, it will be at the application or tenant layer will give an illusion that it is a very big layer to switch, which is connecting all the applications VM of or particular tenant. Similarly, different tenants will have this kind of view.

Now, to translate the physical addresses to the physical addresses to the application addresses, there is a requirement of a indirection or a virtualization layer, which is maintained in the form of a directory server that maps the application level addresses to their to the current locators. So, VL2 has the agents that runs on the server that will query

to the directory service and find out the application address to the location address mapping. And then, when it sends the packet, it will be wrapped the application address in the outer locator header.

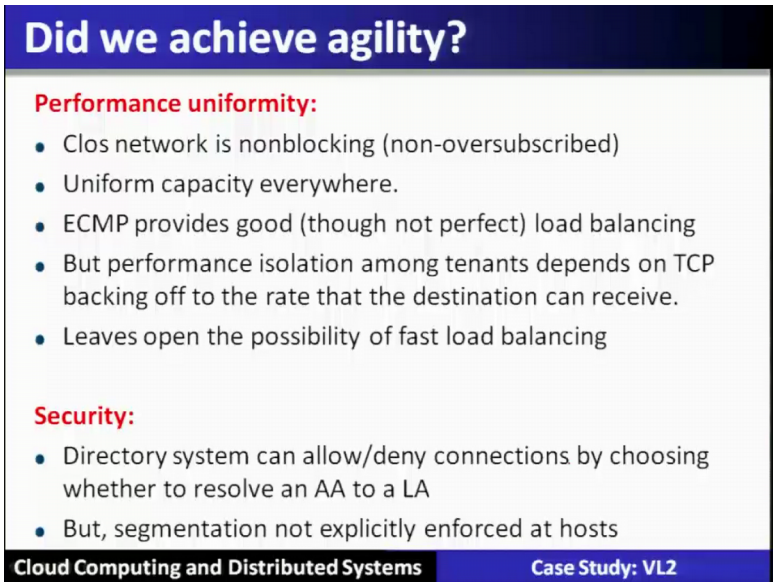
So, it is encapsulation and decapsulation will be done. So, encapsulation that means, when the packet is build, it will be encapsulated. And on the outer header, the application address will be and will be put. Similarly, at the time of delivery, all these decapsulation will be done, and it will be delivered in this particular manner. So, this will support a virtualization layer will support any service, anywhere, that is called agility, that we have already explained.

(Refer Slide Time: 52:45)



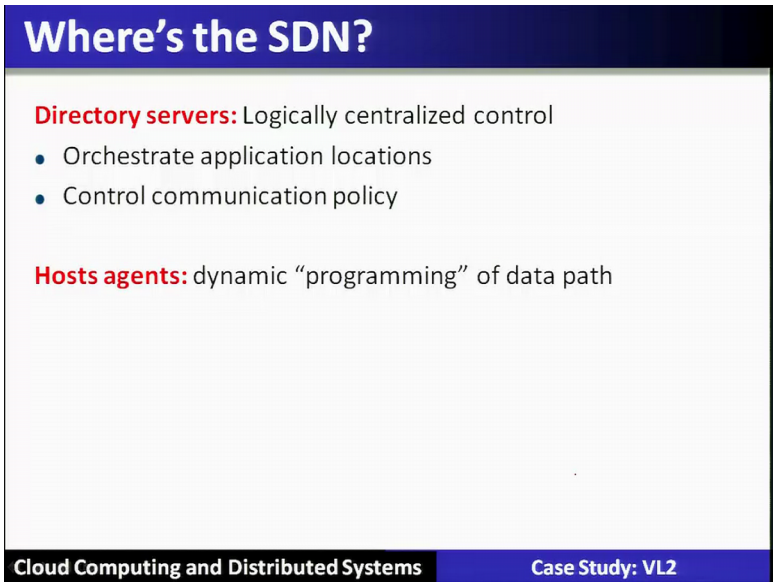

So, did we achieve agility? Let us review this. Now, we have seen that it is a location independent addressing, we have achieved with the help of application addresses. And application addresses are location independent. As far as L2 network semantics are concerned. These agents will intercept and handles the L2 broadcast, multicast, and so on. So, both of these above issues, will require a layer 2.5 that is shim agent running on the host, but the concept transfers to the hypervisor based virtual switch.

(Refer Slide Time: 53:26)



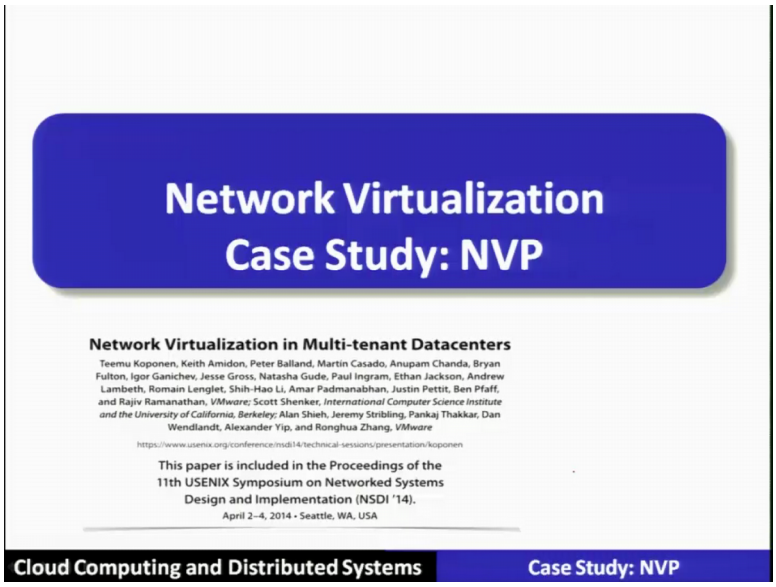

Now, let us see the performance uniformity. So, clock network is non-blocking, and non-oversubscribed. Uniform capacity is everywhere. ECMP provides a good do not perfect load balancing. But, the performance isolation among the tenants depends on the TCP back off rates. And leaves open possibility for the fast load balancing. Security, let us review VL2, whether how it ensure the security. So, directory service can allow and deny connections by choosing whether to resolve an application address to a locator address. So, security is enforced at the level of directory system, but the segmentation not explicitly enforced at hosts.

(Refer Slide Time: 54:16)



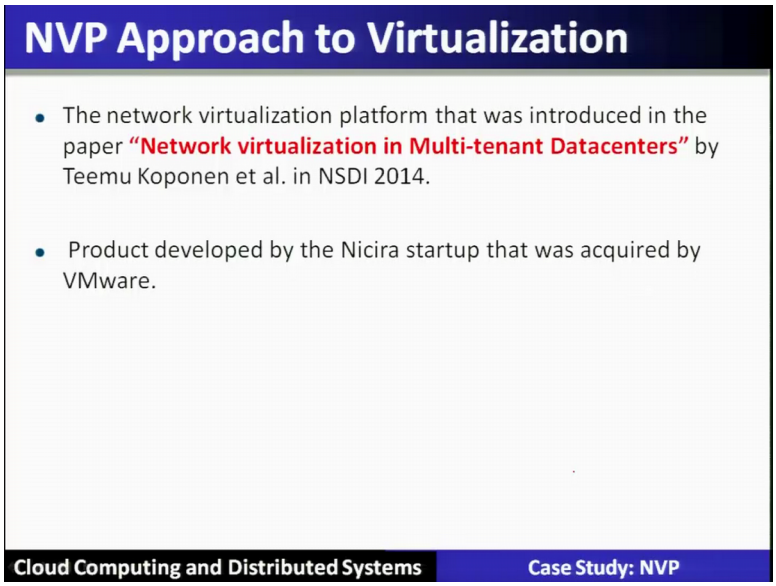
So, here we have seen that software defined networks in the form of director service has been used as the logically centralized control. Why, because it used to deny or accept the new incoming requests of the traffic. So, therefore, the software-defined networks will orchestrate, the application locations will control the communication policies across different directory services. And this host agent will perform the data path, using a dynamic programming approach.

(Refer Slide Time: 55:04)



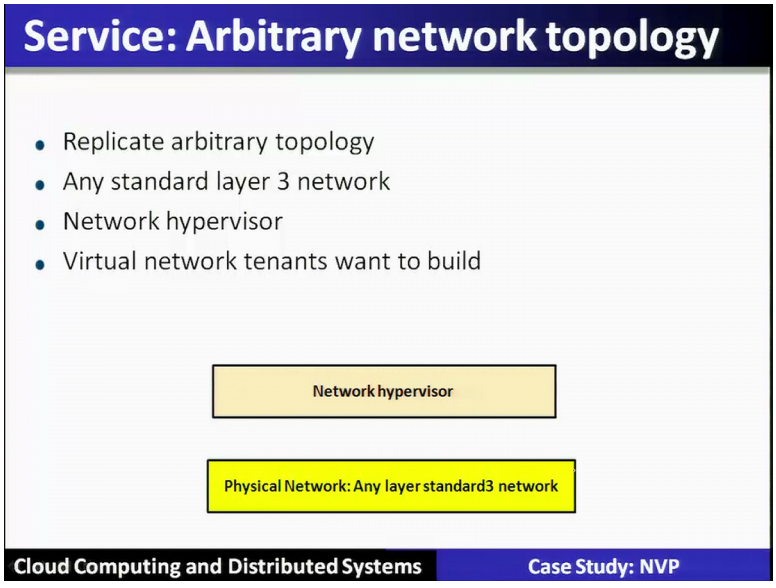

The next case study, the next type of implementation, you will see in the network virtualization using NVP protocol that is network virtualization in multi-tenant datacenters.

(Refer Slide Time: 55:20)




Let us see, how it supports, what is his view. There is another approach to the virtualization of multi-tenant datacenters in the form of NVP approach that is Network Virtualization in Multi-tenant Datacenters, published in 2014. This particular approach later on was taken up by the VMware as a product.

(Refer Slide Time: 55:45)

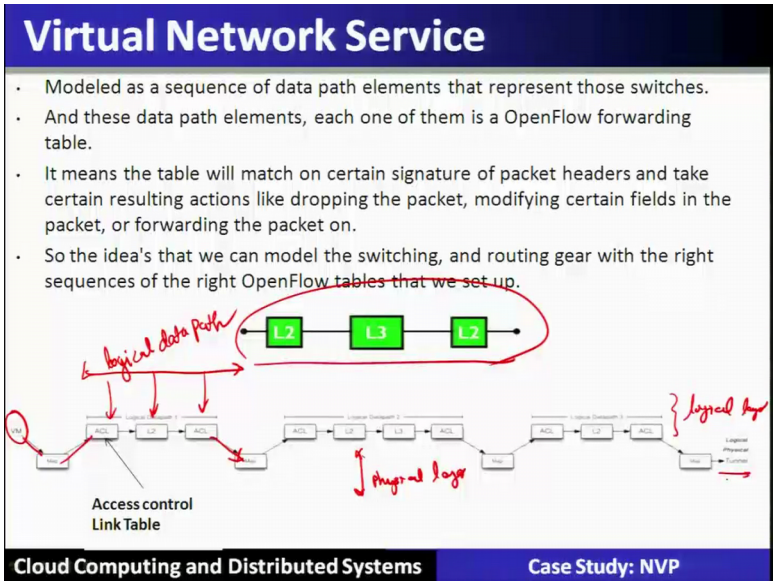



Now, here it will consider for the service any arbitrary network topology that means, it considers the multi tenants to be using different tenants to be using an arbitrary network topology. What do you mean by arbitrary topology, means any combination of layer 2

switch, layer 3 switch, and the networks out of formed out of them is called an arbitrary network topology. It will use a concept of a network hypervisor; just we have seen in the virtualization of the servers that different hypervisors are there that is similarly, here the network hypervisors will be introduced for the first time.

So, therefore, what this NVP provides, it is provides a virtual network for the tenants, the way they want to build. Let us see, how it does. So, we have shown over here that the below is the physical network, which can be any standard layer 3 network, which is being virtualized that is called network hypervisor.

(Refer Slide Time: 57:15)

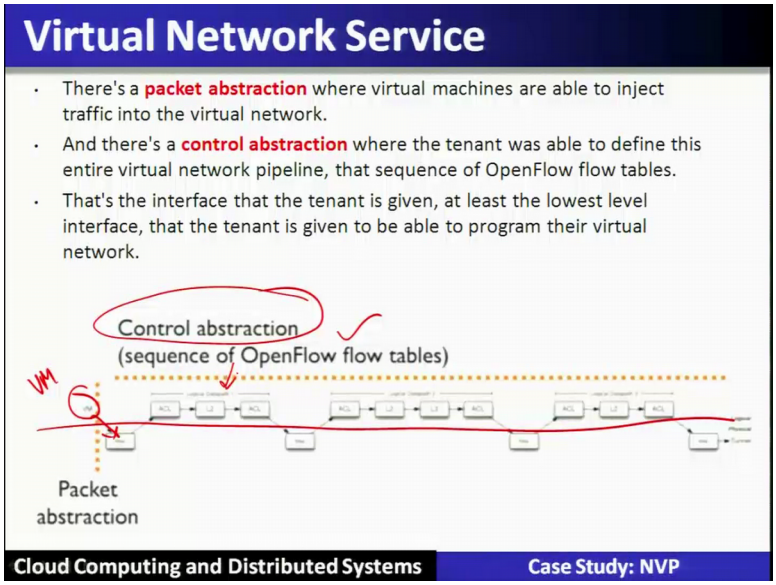



And above that, you will see a network virtual network service; so, modeled as the sequence of data path element that represents these switches. And these data path element, each one of them is OpenFlow forwarding table. So, it means that table will match on a certain signature of the packet headers and take certain resulting actions like dropping the packet, modifying certain fields, and so on. So, using open flow tables, you can form the network a virtual network in this way, which is shown as L2, L3, and so on.

So, this particular virtual network service, we can understand here. If there is a virtual machine will be mapped, which intern will mapped to a particular virtual network in the form of a logical data path that is nothing but an access control link table, followed by an L2 switch, followed by another access control table. And this will form a logical data path. And then, again it will come for the mapping to the physical layer. So, this is called

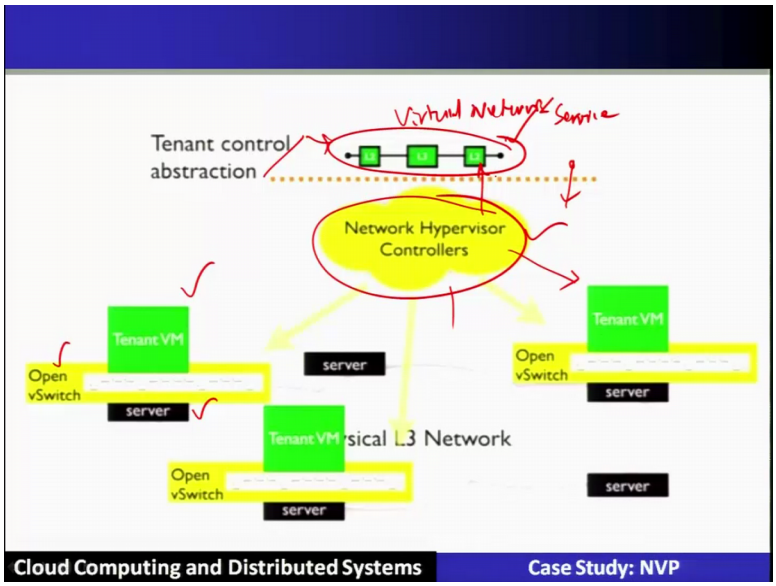
physical layer. And this is the logical layer. So, the virtual network service can be provided in this particular form of a logical layer. And finally, this particular or a packet will be send through the tunnel.

(Refer Slide Time: 59:23)




Now, here there will be a separation that there is a packet abstraction, where the virtual machines are able to inject the traffic into the virtual network that we have shown. And there is a control abstraction. So, control abstraction that is let us see this view. So, virtual machine can inject the packet, and the packet abstraction will be that the control abstraction will be provided in the form of a OpenFlow tables that we have seen in access control link tables. So, that is the interface that the tenant is given, at least the lowest level interface that the tenant is given to be able to program their virtual network.

(Refer Slide Time: 60:28)

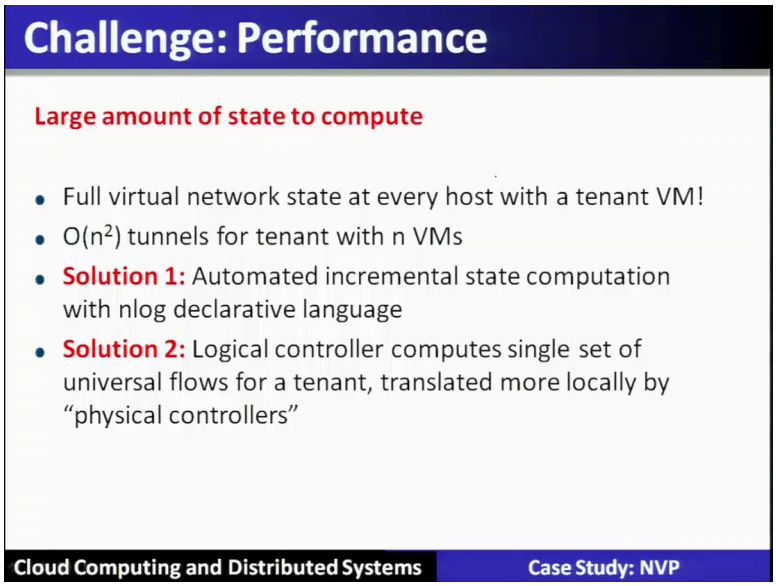



So, let us see the virtualization in more detailed of the virtual network. So, this is the abstraction, which tenant can see its virtual network. So, this can be formed or tenant can specify its network in this form of virtual connections. And the tenant can control it over this particular topology. Now, this particular control topology or control abstraction will be supported by the network hypervisor controllers. And they will manage all the tenant virtual machines.

And this tenant virtual machine is being supported by an Open vSwitch, which manages the server. So, just see that, these network hypervisor controllers will provide an abstraction to the tenants in the form of a virtual network, which intern will manage the tenant virtual machine. The tenant virtual machine in turn will be supported by the Open vSwitch. And Open vSwitch manages the servers.

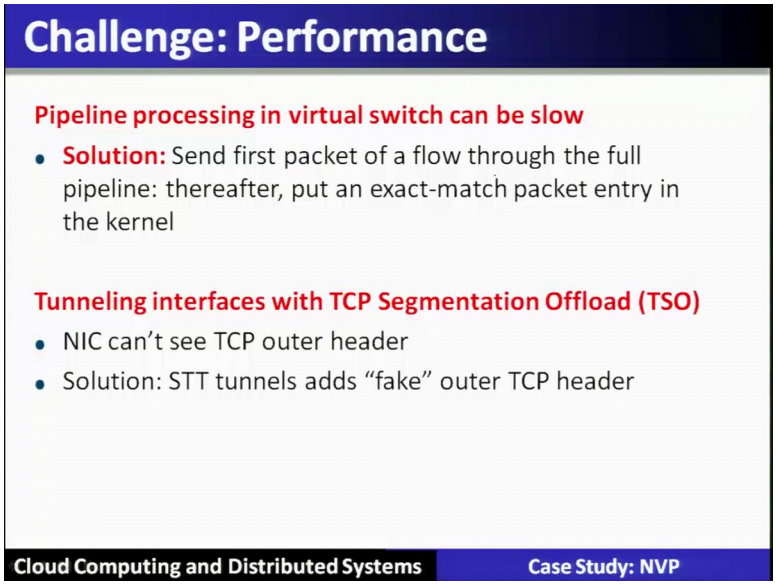
So, Open vSwitch is managing the devices and gives the support to the tenant virtual machine. And the network hypervisor will support to the tenant control to the tenant virtual network. So, this way every tenant can establish or can be specified through the abstraction its virtual network which in turn is virtualized by a network hypervisor, and using vSwitch all the network devices are being managed. So, therefore, NVP has given the idea of the network virtualization. So, it has given the concept of network hypervisor, just like we have seen the hypervisors or virtualization of the servers.

(Refer Slide Time: 63:21)



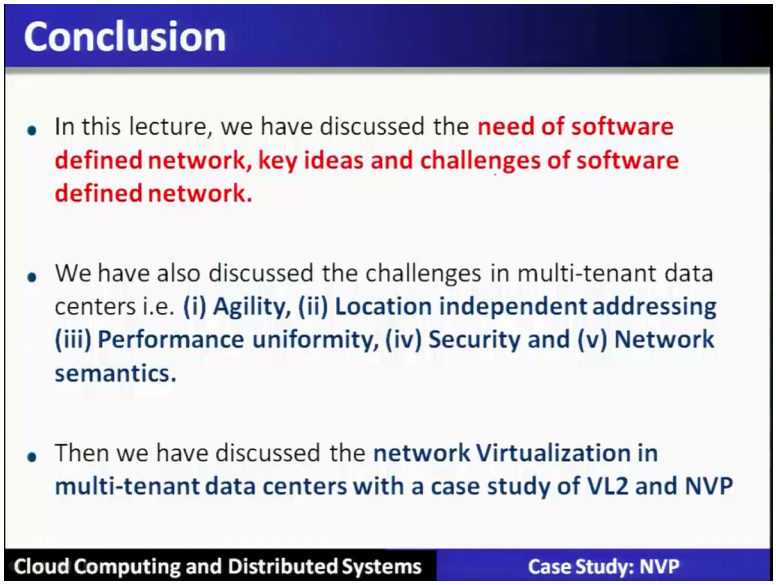

Now, the challenge is let us see the performance. Here it requires large amount of state to be computed. Full virtual network state at every host with a tenant virtual machine, it requires order n square different tunnels for tenants with n virtual machines. So, the 1st solution says that automatic incremental state computation with nlog declarative languages is possible. 2nd solution says that logical controller computes single set of universal flows for a tenant, translated locally by physical controllers.

(Refer Slide Time: 64:06)



So, therefore, the performance can be handled through the different programming of pipeline processing in the virtual switch can be slow. So, it is a tunnel interface with TCP offloading can be there.

(Refer Slide Time: 64:24)




So, conclusion, in this lecture, we have discuss the need of software-defined networks, and how this software-defined network has overcome from different challenges of the traditional network. We have also seen the use of software-defined network for virtualization in a multitenant datacenters to support the agility, location independent addressing, performance uniformity, security and network semantics. We have then discussed there are two different ways of network virtualization in multi-tenant datacenters in the form of VL2, and another way of design that is called NVP.

Thank you.