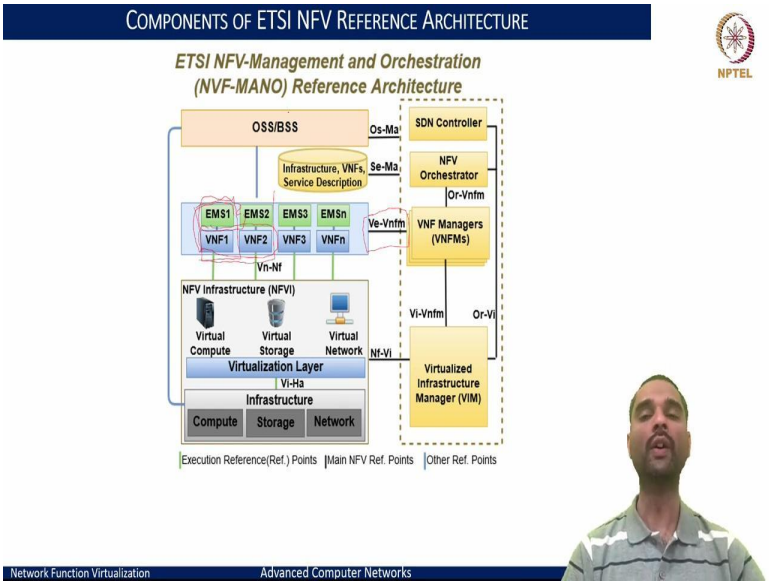
Advanced Computer Networks
Professor Doctor Sameer Kulkarni
Department of Computer Science Engineering
Indian Institute of Technology Gandhinagar
Lecture 40
Network Function Virtualization - Concepts, Framework and Architecture - II

(Refer Slide Time: 0:19)



So, we looked at the NFV framework that enables dynamic construction and management of the virtualized network functions. Now let us look at the key architectural components of this framework and what are all the key abstractions and interfaces that would be necessary to realize such a framework. And in this, ETSI came up with what we call as the ETSI NFV reference architecture also known as the NFV MANO or the ETSI NFV Management and Orchestration Reference Architecture.

And again the components that we saw earlier to the components that we see now are more or less the same, but there are some additional components that we need to discuss in detail. First, this NFV architecture that was proposed by ETSI has, in fact, helped define the standards for NFV implementation and so we need to understand what are those standards and it would not be possible to cover all of them to a greater depth, but at least we should try to understand what are the main architectural reference points and what are the executional reference points that are being leveraged to build this reference architecture.

And in essence, this NFV reference architecture consists of, at the core, the virtualized network functions or the software applications that would be deployed to deliver the desired network functions on the NFV infrastructure. So, we looked at NFV infrastructure encompassing a virtual storage, compute and virtual network, including the hardware infrastructure of compute, storage, and network.

But when we think of these virtual network functions that are going to be deployed, then we need to ensure that these network functions are managed properly and to do that, ETSI came up with what we call as the EMS or the element management systems, which would be responsible for the functional management of these virtual network functions.

And in this, by management, what we mean is to include the functions of fault management, the configurations that we need to set for VNFs, what is the core required, the compute requirement, storage requirements, memory requirements, all of these configurations that you would want to do, the performance monitoring, what are the performance characteristics that you are looking for, what are the SLAs that this VNF needs to be supporting and what are the security constraints in terms of access control, authorization aspects, all of these configurations can be plumbed into what we call as the element management systems.

And note that these element management systems can themselves be deployed as virtual machines or virtual instances that would run on the same infrastructure, but this would provide us the handle to control and configure the network appliances that we would want to run. Although in this figure, what we have shown is that there is one-to-one mapping for an element management system to the VNFs, this is not necessarily true. We may have a single EMS which is able to be managing multiple of the VNFs. So, one-to-many is possible in terms of multiple virtual network functions being managed by a single EMS instance.

And now, these EMF instances or virtual network functions need to be managed from the network operator's point of view. And for a network operator, you would want to have control from one place to say how these configurations can be done. So, for that, this interface of virtual elements to the virtual network functions, the MANO infrastructure that we would want to facilitate needs to be given, and this interface is between the core layer of what we saw as the MANO layer.

If within this, we need specific functions, what we call the virtual network function managers, which would manage the subset of these network functions because one of the kinds of network functions can be deployed by some software vendor while the other kinds of network functions may be coming from other kinds of a software vendor. So, we may have different virtual network function managers to manage all of these different kinds of virtual network functions. So, in essence, this VNF manager is responsible for the life cycle management of these virtual network functions and the configurations on the EMS.

And when we mean by life cycle, is when to instantiate this network function, when to have the software updates for these network functions, how to version these network functions and ensure that you can monitor the performance levels, diagnose defaults, and if there are any failures, how to heal or ensure that they can be respawned and when their life cycle is over how to terminate these virtual network functions. All of these become the key aspects for this virtual network function managers to handle.

And once we have provided these virtual network functions to the virtual network function managers interface, that would enable any open environment so that multiple managers can be plumbed in on a single infrastructure. Second, like we have at the bottom, is the network function virtualization infrastructure which encompasses both the virtual and the physical infrastructures; this again when we want to be controlled or like when we have a premise or a campus or enterprise we have these infrastructures that are going to be set up.

We need a handle to again manage this infrastructure, and that is exactly the role of the virtual network infrastructure manager or what is being termed as the VIM here in this diagram. And this virtual infrastructure manager is responsible for managing the compute, storage, and network resources, software or virtualized infrastructure, as well as the physical infrastructure. Then what we would also need is to see how these network functions can be brought to life in terms of managing the life cycle; there have to be some policies on which you would want to act.

And that is facilitated through the component of what we call as the NFV orchestrator in the MANO. And this NFV orchestrator is basically the one that automates the deployment, the operation, the management, the coordination of the VNFs when we spoke about how to chain these VNFs, what are the means to connect these resources, all of this could be coordinated from

this NFV orchestrator. So, this NFV orchestrator basically covers the orchestration and life cycle management of the physical as well as the software resources that are needed to run these virtual network functions.

So, that means this NFV orchestrator would need to communicate with the virtualized infrastructure manager on one end and also need to communicate with the VNF managers to spawn or create the necessary virtual network functions and set up the configurations, etc on them. Thus this NFV management and orchestration focuses primarily on all virtualization-specific management tasks that are necessary to run this NFV framework.

And here, we can also see that there is a role of an SDN controller. Although the placement of an SDN controller in terms of the architecture of NFV was highly debated SDN controller, like we saw in our earlier lectures, is to manage how we want to set up the communication amongst these network instances to manage or provision the resources or these virtual computes that we would want to build in this NFVI.

So, we can think of an SDN controller as working in conjunction with the NFV orchestrator to set up the forwarding rules that we want, what control functions that we want to build for these, what is the topology that we want to build with the virtual links, all of these aspects can be managed by the SDN controller which we can think of is being situated along the lines of NFV orchestrator and the NFV manual framework. And the other most critical piece that any of the operations in telecommunication or enterprise would have is what we are showing here as the OSS and BSS component.

And in OSS and BSS what we mean by OSS is the operations support system and what we mean by BSS is the business support system. Think of the operation support system as a means to facilitate communication, to facilitate the concerns of whatever we need to deal from an operator point of view to manage all of these resources.

And from this end, what we would see is basically that the means to maintain and operate all of these devices, we need a specific database where the things would be basically stored in terms of what information we are gathering about the infrastructure, gathering about the services that we want to build and operationalize these services. And think of this operationalizing as taking care of the entire infrastructure that we have and to ensure that we meet the utility over this

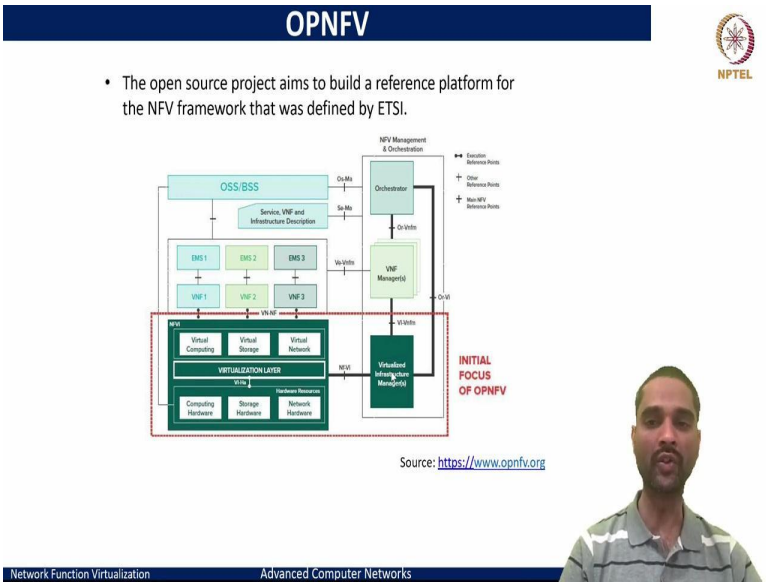
infrastructure. Like when we have a telecom operator who is trying to onboard a new user, what is his status that is being maintained for a given user on the local premise is managed by the operations support system.

And the other aspect is the BSS which primarily consists of how to interface with the customer or the end user. Like if there is a customer relation management information that needs to be set up, like when we talk over our mobiles or when we use the data, there is a billing that happens. So, how these telecommunication billings that need to be done is given and updated to the user, all of these aspects come into the business support. So, think of it in very simple terms if we have to say the interface, for example, between the BSS is for capturing the user's requests or capturing the order from the end user; like user specifies the intent to the networks through this BSS framework and how you realize such an intent of end user, like the user wants to have 100 GB data that he wants to purchase that is done through the BSS support system. Now once that is done, how it is being operationalized and ensuring that the requests of a user are fulfilled is what governs what we call the operations support systems role to fill in and do this job. And this is where like the reference points of this OSS and BSS, how they interact with the virtual network functions and how they interact with the virtual network infrastructure hardware have been categorized as other reference points for communication.

Note that the communications that are marked in green are the execution reference points for the VNFs to interact with the hardware and, likewise, for the virtual layer to interact with the infrastructure, that is, the software abstractions that we are trying to build for the VNFs to work on a given infrastructure and the main NFV reference points or the ones that are marked here in black are the ones that allow us to manage and control such an infrastructure and build the desired services over this infrastructure.

And whenever we have such infrastructure requirements, we need to catalog a lot of metadata. That means there is an implicit requirement to support the database where we catalog the infrastructure, the kinds of network functions, the service descriptions, all of this data in one fashion this could be a distributed framework by itself. Overall this was how the ETSI NFV reference architecture was delivered from the ETSI group.

(Refer Slide Time: 13:29)



The ETSI NFV reference architecture saw various open source implementations, and we will look at just one of the open-source implementations known as the OPNFV, which it stands for the open platform for NFV and this is a collaborative project under the Linux foundations and that has helped transform the global networks through this open source NFV.

And what this OPNFV tries to do is to facilitate the real implementations of the reference points that we discussed in the earlier framework or the reference architecture, and it initially started with the focus on the virtualized infrastructure manager implementations of how to communicate or bridge this interface between the virtual infrastructure manager and the network function virtualization infrastructure as a key piece.

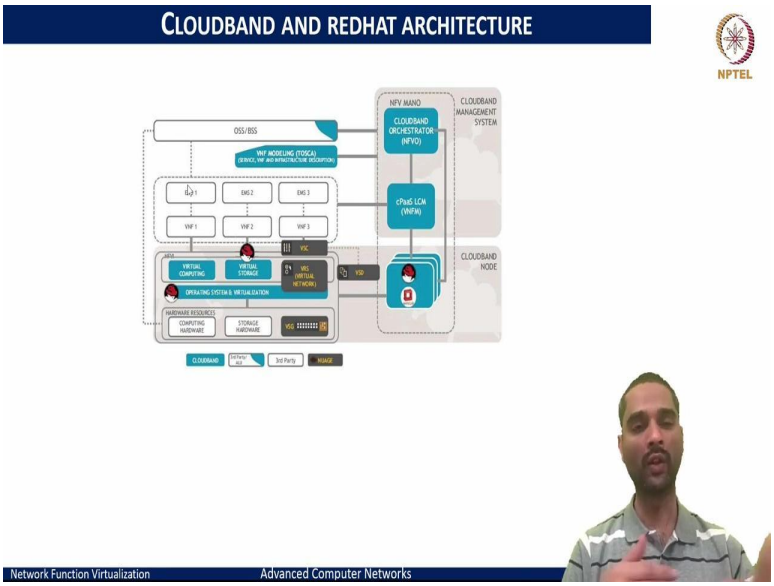
And then it started to build all the other components and this OPNFV, which facilitates the common NFVI and also facilitated a through the project as a continuous integration with several of the upstream projects and when we say what were the core upstream projects that we see in this architecture where they were having the means to develop and deploy network functions and primarily it is the open stack that was there already to create VMs orchestrate and where to deploy the VMs.

So, this OPNFV was trying to support integrating this NFV framework which is by developing the VIM and the NFVI with integrating them with the upstream projects like open stack, and when we think of the containers, we could have the Kubernetes as a framework that would

support and when we think of the controllers, SDN controller they service to support the open daylight and many other like ONOS and so on.

So, the overall goal was to build this reference platform that any of the telecommunication operators or even the enterprise operators could take and start to build the NFs and work them out. And this has been a huge success, and it is continuing; although it started with just the bottommost layer, it has now grown to address the VNF manager, the orchestrator, and the interfaces between these points.

(Refer Slide Time: 16:22)

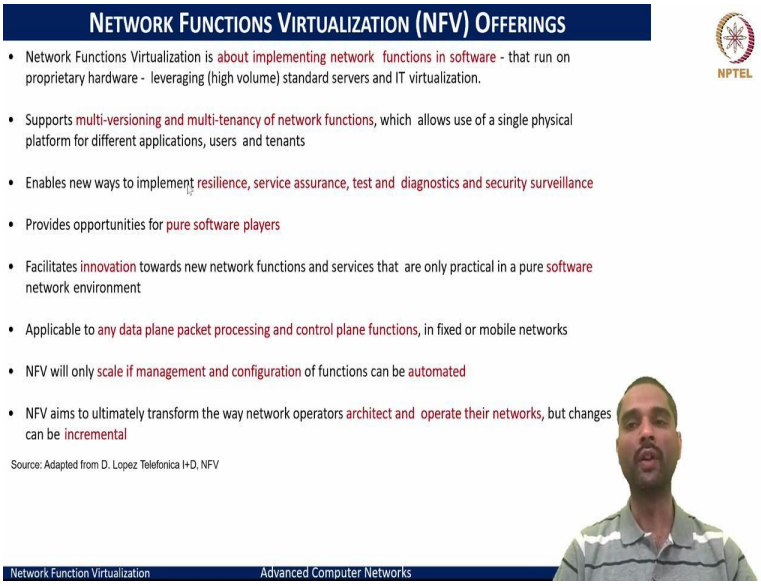


Further, there is also this Cloudband and the RedHat architecture. This is infrastructure based on how the RedHat Enterprise Linux open stack integration could be done with the Cloudband, which was an Alcatel-Lucent-based NFV platform, and this, in a way, offers a new way to design, deploy and manage the network services. You can think of having the RedHat open stack, which is now facilitating how to manage and provide control over the NFVI and implement the NFV manager and also provide the Cloudband orchestrator communication with that to build the VNF models.

And TOSCA is one of the core protocols or mechanisms to see how the NFs can be modeled to build the abstraction, the logical abstractions of providing the services that we want to build for realizing these network services. And these have been very prominent and successful, and there

were many more of these architectures that have been instantiated, but we need to see, we have this framework architecture, and we have so many of the reference implementations, but what exactly do they offer in terms of what are they bringing to the table?

(Refer Slide Time: 17:51)



And as we know, this NFV is about implementing the network functions in software, which would now run on generic hardware. And in order to do this, we need to say when we create a framework to instantiate these network functions, we are thinking of a data center or a cloud-like environment where multiple users are using the same resources. That means we need to ensure that the framework implementations that we have are able to support multi-tenancy.

And what we mean by multi-tenancy is basically an architecture where a single instance of software application or the hardware that we have would serve multiple customers. And each customer is what we refer to as a tenant. And in this reference, the mode of operation for the software where multiple independent instances of the same software could be run by different tenants. And we still need to ensure the isolation for the software infrastructure, but they may still be using the underlying same shared environment.

So, logical isolation amongst multiple instances of the same software that we are going to run is to be guaranteed. And this is where the NFV really helps by ensuring that we are able to decouple the hardware from the softwarized instances. And each of these softwarized instances

can now be controlled and configured independently by different vendors. And that is how it supports multi-tenancy. And another important aspect, whenever we try to build systems, we evolve. And when we evolve, we start versioning the systems.

And then, when we expose these to the public, there may be a need where one user would want to operate on a particular version while another user would want to operate with a different version. So, there is also a need to support what we call multi-versioning, wherein which allows the network functions of different versions to coexist on the same framework. And this makes it also simple. Whenever we want to have upgrades or have to switch back to a stable version when something experimental is not working, we could have the previous version rolled back much more simply rather than having to restore the entire infrastructure.

And thus, this NFV, through the use of VMs and the right abstractions, enables to support these cases. Further, this NFV also, we discussed saying that it would allow for having some sort of resilience and service assurance and also ensuring that there is a security diagnostics and surveillance that we could do with these functions. So, the reference implementations that we want to build and the architecture need to have the abstractions or the APIs to facilitate these information.

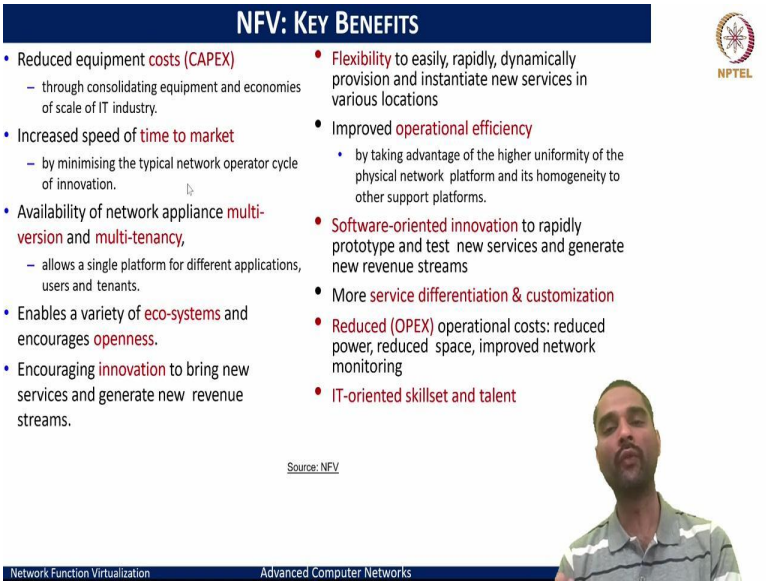
And overall, when we have such a framework, what it really creates is the opportunity for many of the software players to come together and independently even build different network functions. And this is where the innovation starts to foster in terms of how different software frameworks can be pulled together and create new services, and even collaborate with different network functions to create multiples of these services to work.

And in doing all of this, now what this NFV really tries to facilitate is to provide a scale for deploying a large number of instances, meet the demands of the traffic load to scale out or scale in the required instances of the network functions that we would want to run. And all of it can be automated in terms of trying to bring a new instance of network function, managing its life cycle, scaling the resources, or even scaling up the resources of CPU or memory requirements for a particular function can all be done in an automated fashion.

And thus, what NFV actually offers is to ultimately transform the way we can think of the role that the network operators have to do in terms of building the network infrastructure. But note

that this often always has to be incremental. We cannot have a zero-day where we shift to this NFVI. So, they may still have to deal with the legacy hardware, but when it comes to the NFV infrastructure, they may be able to automate many of it and mix and match in their infrastructure as they see fit.

(Refer Slide Time: 23:08)



Thus, what this NFV essentially offers is in terms of economy, it reduces capital expenses and ensures that we can run multiple network functions on the same hardware, that is, through the consolidation of equipments, which otherwise would have been, you will have to purchase proprietary and different kinds of equipments.

Second, it also helps enable the speed for the time to market. And that is where the acceleration for innovation has been really upped by NFV, wherein we minimize the time that the network operator would need to deploy, prototype, and idea because he is no more dependent on the hardware operation equipment manufacturers.

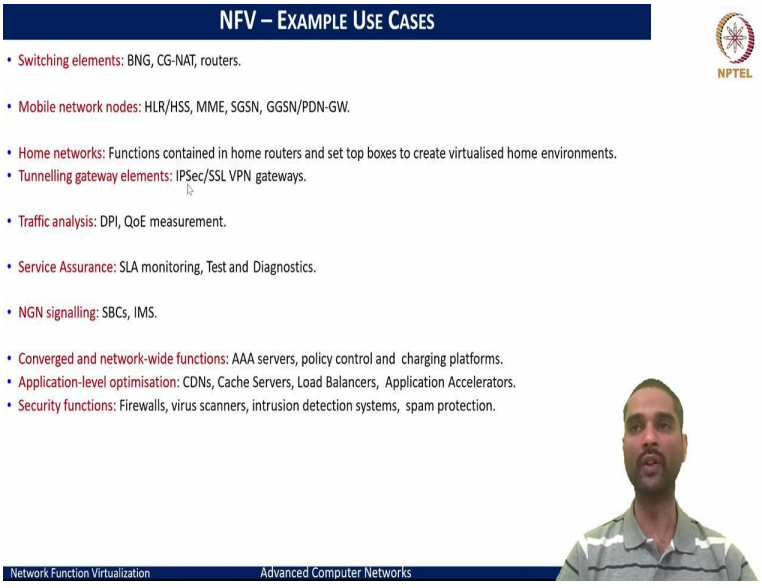
Third, through the support of multi-versioning and multi-tenancy, we are able to now allow many of the platforms to coexist on a single infrastructure. So, that is a consolidation of multiple infrastructures as well. So, it is again adding to the economy of scale, saving on a lot of the aspects of capital expenses, and also, from the user point of view, it also saves a lot of resources and money.

And overall, this has enabled the open ecosystem for innovation and also provided the flexibility to the network operators to easily manage, readily deploy, and provision new kinds of resources and also provide better operational efficiency, wherein now they can take advantage of the uniformity in how you want to manage the resources, you no more need to dig into the different specs of different devices to just manage the management is one point.

Although the configurations and updates may require you to deal with the specifics, but in terms of the operational management, it would have been a lot more simpler. And overall, what NFV has tried to do is provide software-oriented network functions which are open to innovation, which we can readily, rapidly prototype and deploy and test out. And in a way, we have basically de-ossified the network infrastructure and enabled the network functions to take over and softwareize these networks.

And all of this also accounts to a lot of operational expenses minimized due to the fact that we do not have to have specialized labor to spend on. We have less of a power. You are not having multiple of instances that would run for different purposes, but just commodity hardware running multiple instances; lower requirements on the space to host and deploy these instances and also in terms of monitoring now, the number of devices have gradually shrunk. So, you will also be able to monitor these things very easily.

(Refer Slide Time: 27:00)



And all of this, in a way, is good, but it also means that we have now, in a way, tried to cap off on how the IT-oriented skillset and talent can be reused for managing the NFVs because they are now built on the commodity hardwares. And we can also see it as an augmentation that it is trying to bring for the IT skillset to operate on the NFVs and build these NFVs very easily.

So, this NFV supports a lot of use cases, like we said. And if we have to start with the basic switching elements, that is basically in the telecommunication operator's world, we can have a broadband network gateway or the BNG, which is basically the access point for the subscribers through which they connect to the broadband network.

This can be deployed as a virtualized network function. Carrier-grade NATs that we also discussed earlier can now be thought of not as dedicated hardware but as just the network functions that we can deploy on the commodity hardware. And more so, we can even deploy just the routers as the instances that we can run on the commodity hardware as just the software, like Quagga that the Linux world offers can be used to run the different router configurations that we want to build.

And when we see from the mobile network operator's point of view in the mobile network infrastructure, we can implement basically what we call as the HLR and HSS modules, which are basically the home subscriber servers, which is basically in the 4G kind of a network where the main function that they would provide is to facilitate the communication of the network with the subscribers' profile and authentication information's for a 4G user. And likewise, maintain this information as a software instance rather than as dedicated hardware.

And we could also have what we can think of as the GPRS nodes, SGSN, or the serving GPRS support nodes, which are used for basically ensuring that all packets switched data within the network is able to operate and authenticate the users. And this mobility management, all of these specific functions, which were earlier implemented as proprietary hardware can now be implemented as just the software. And likewise, the SGSN or the serving GPRS support node, which could also be, can be implemented as a software instance.

We can also have the GGSN or the gateway GPRS support node, which basically is responsible for the internet working between the GPRS network and the external packet-switched networks like the internet to speak of, wherein they keep a record of the active mobile users who are

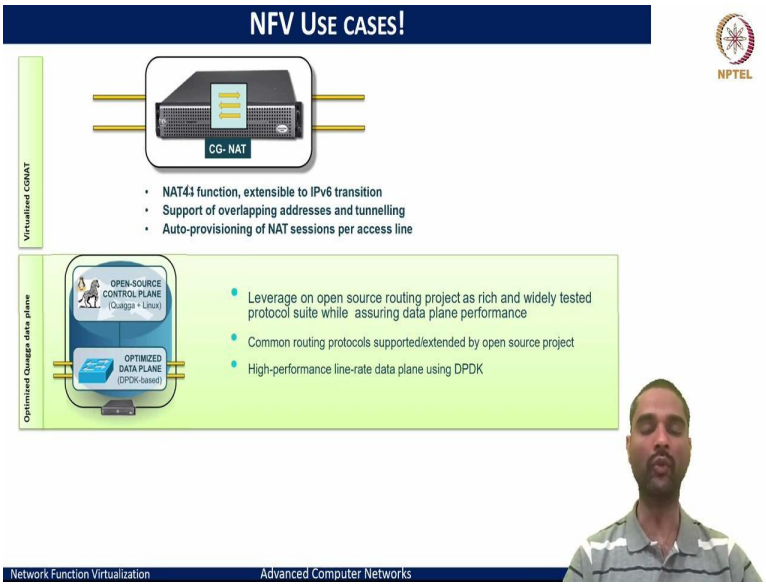
attached to the SGSN network and facilitate for the mobility of these mobile users. So, all of these functions, which were earlier the proprietary hardware with custom protocols that were being built, can now be built as software functions.

Likewise, the packet data network gateways, which are the critical network functions in the 4G mobile core network or the evolved packet core EPC networks. Hence, we can see that NFV can be applied for ISP networks, for telecommunication networks, including home networks where the ISPs can run specific functions as whatever this gateway devices that we would use to hook to the internet can now be seen as the virtualized instances where they can provide numerous home services for home environments. And in fact, when we connect with enterprise networks, we typically use VPN gateways or IPSec-based VPNs.

Now, all of these functions can be implemented in the enterprise network as the virtualized network functions rather than as dedicated hardware, which cost in several hundreds of thousands of dollars. Also, for any kind of network, be it data centers, be it enterprise or telecommunication, if we want to do any traffic analysis like DPI or quality of end-user experiences, all of these measurements that we have to do, they can be built as custom functions and not necessarily have to be dedicated hardware, including monitoring for the SLAs, diagnostics that we want to run all of them.

So, in a nutshell, we can see the NFV replacing many of the custom or proprietary hardware that otherwise were deployed in any kind of network, including firewalls, virus scanners, IDS, spam protections, and so on. So, all of these become the key use cases for NFV, where they can really transform the way we operate with the networks.

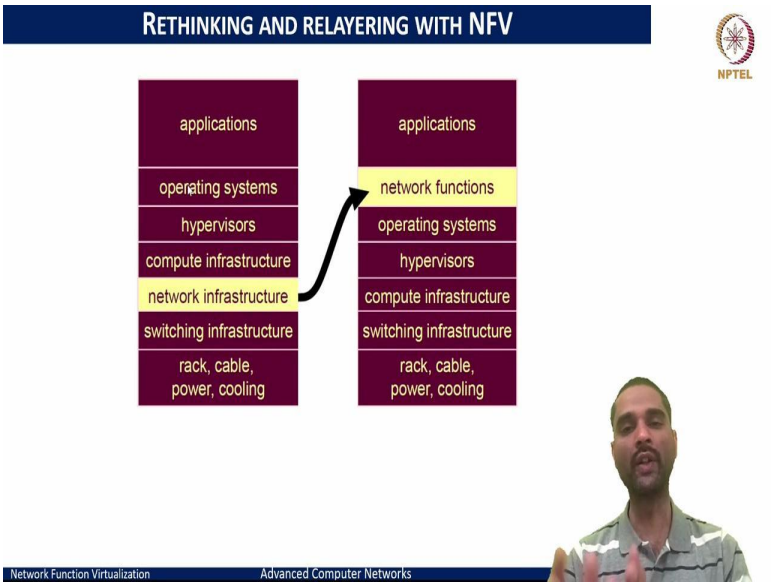
(Refer Slide Time: 32:16)



In essence, like if we look at this NFV use case as a carrier-grade NAT, we can build the NAT44 function, which is a software function that would manage a table, which would keep the lookup of the public and private IP and manage the setup for translation within the IPv4. And also, it could be the same device that we can now use when we want to tunnel and use the IPv4 and v6 devices. So, we can have a NAT4 address inside while we can represent a NAT6 or IPv6 address to the outside world. And we can even facilitate overlapping addresses in a much more easier fashion.

And if we see that a NAT is becoming a bottleneck, we can instantiate a new instance of a VM to take over and enable that we can scale elastically the kind of functions and run them in our hardware. And likewise, when I mentioned about the routers, we can think of open-source control plane that we can build using the Quagga and Linux framework. And we could also build optimized data planes to facilitate more performance-oriented operations, which we will look at the data plane DPDK in the next lecture.

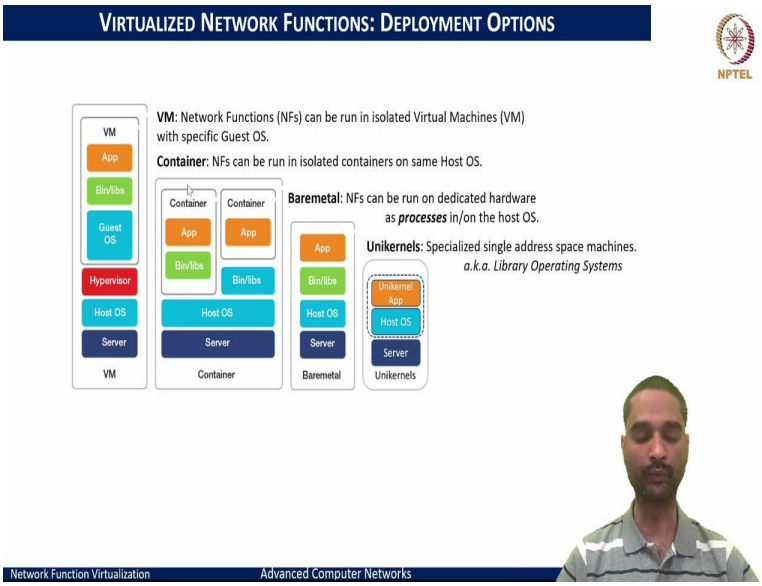
(Refer Slide Time: 33:42)



Nonetheless, what we have really tried to achieve when we rethink of the network infrastructure that used to be around prior to NFV, what we would see is at the bottom, we have the hardware infrastructure, which consists of the rack, cable, power, and cooling to run the hardware infrastructure consisting of the compute nodes or compute hardware's, x86 machines or any ARM machines or any of the servers, the switching infrastructures like routers, the switch devices, and the network infrastructure that encompasses variety of the middleboxes, which used to be the core at the hardware.

And on top of these, we used to have the hypervisors, OS and applications run. Now, we have taken out these network infrastructures and moved them into the software layer. That is, now you can think of the infrastructure as just the compute and switching infrastructure that is there as hardware. And all the network infrastructure, which were the hardwares earlier, have now moved to a software network functions that would run on top of the same compute infrastructure. And this shift is exactly what NFV has enabled and the shift has also led to a lot of innovations that we can think.

(Refer Slide Time: 35:11)



So, now, when we see these network functions, the next immediate question would be, how do we deploy them? And what we have started earlier was the dedicated hardwares. Now, we want to run these, get rid of these dedicated hardwares and run on top of the commodity hardware.

So, one of the most simplest approach would be we have this dedicated commodity server, which is running a server host OS and has comes with associated binary and libraries. And we deployed a network function as a process that would run. So, to consider this as a Linux box and I am running an application like Quagga on top of this Linux box. So, that is one of the most simplest implementations that we can have.

So, here, we are running the network functions directly on top of the physical hardware. But with NFV, we want to bring in the virtualization and with the virtualization, what we can think of as a most common means to deploy is as a traditional virtual machines. And here, what we mean is we create a virtual machine on top of the physical infrastructure through the use of the hypervisor and run our applications or network functions as an application within this virtual machine, which would have its own guest OS that would run on top of the hypervisor.

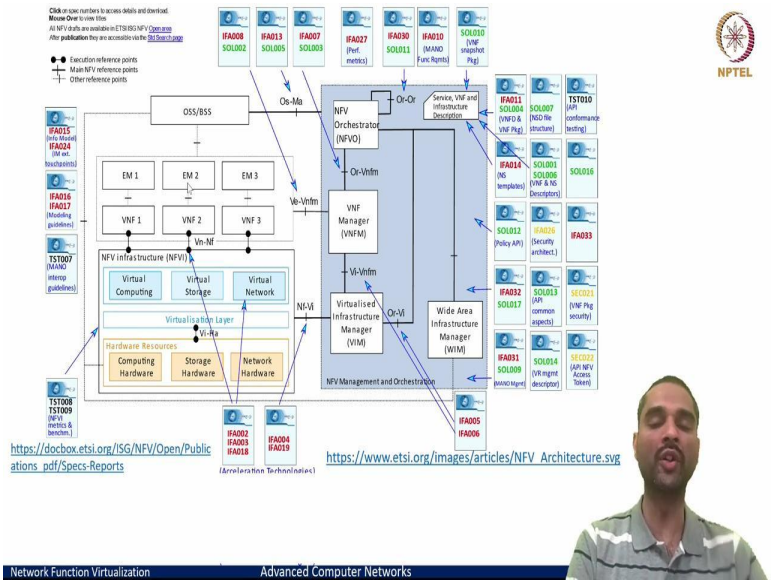
And this facilitates at most isolation, the maximum isolation and ensures that these virtual machines can be spawned and run easily. But these are in a way heavy-weight because we are now trying to maintain a guest OS, the binaries that are needed for that virtual machine to be run on a single hardware. And they are more intensive in terms of the memory and storage. And what we have seen as a shift in this virtual machine model is the lightweight containers.

So, we could even build our NFV infrastructure, the virtualized infrastructure to run these containers, wherein NFs can be thought of as the applications that are run within a container. And the container as a scope provides the isolation for that particular application, but it would reuse the same underlying host OS. And that way, these are much more lightweight, and we are able to deploy these containers in a much more faster way. That means the time to start or time to spawn a new instance would also be greatly reduced when it comes to the container model of running these network functions.

And there was also an interesting path that the researchers proposed and what we call as the unikernels. Here, think of unikernel as a dedicated kernel which is just doing one function at a time and that is exactly our network function. So, the unikernel app and host OS are basically the drilled down or a very thin or lightweight host OS and app which just do the function that we want. So, think of these as what we can see, when we want to run very specific code and just do a particular job alone. And because these are now softwarized in terms of what we would want to run, we can run much more of these unikernels on the same server so that we are able to scale a lot more than what we would be able to scale or with VM end containers given the limited resource constraints that we may have.

And these also unikernels make it a single address space wherein if something fails, it is easier to respond to another instance or update the state much more easily. And these unikernels are also called as library operating systems. And one of the most celebrated works in this is the Qlik OS as a means to develop the network functions using the Qlik OS infrastructure as a unikernel.

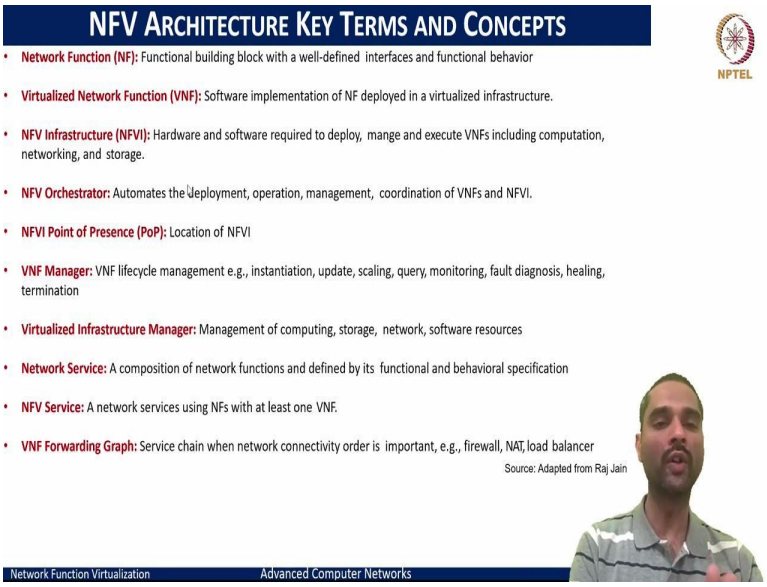
(Refer Slide Time: 39:35)



To summarize, what we have seen is the evolution of the NFV framework and the reference architecture. And for each of the reference architecture, as it evolves, we need standardized definitions of what interfaces we are going to operate. And the slide here precisely shows what are all the ETSI NFV groups, ISGs, and deliverables which have enabled us to build these abstractions and build these reference points for implementing the open standards.

And you can see that we started with just six of these specifications and documents, but these have culminated to more than over 30 odd documents that have come up trying to foster the openness and innovation in this NFV infrastructure. I have put the links for those to look at, and in order to access these documents in the ETSI, all of these are in the public domain, and that would give us a good hint of what the way the evolutions have happened for this NFV just over the last 10 year.

(Refer Slide Time: 40:46)



So, NFV, if we have to understand to the core, we will have to deal with a lot of terms and concepts to deal with. I have tried to cover some of them and address them in earlier lectures, but some of them we may not have covered. So, I have tried to put some of the key terminologies and the concepts that are associated with understanding the NFV for reference in these slides, including the most relevant acronyms that you may see in some of these slides and that you may often encounter in trying to understand NFV.

So, next up, what we will try to do is we will look into the key challenges. We have looked at how these benefits have evolved from these network functions, but any technology like when we

bring it also comes with its own sets of challenges. So, it is also important for us to understand the key challenges that come along with this development and deployment, and the realization of the NFV framework. And we will try to do that in the next class.