

Advanced Computer Networks
Professor Dr. Sameer Kulkarni
Department of Computer Science Engineering
Indian Institute of Technology, Gandhinagar
Lecture 43
Summary and Comparison of NFV and SDN

(Refer Slide Time: 0:19)

SO, WHY WE NEED/WANT NFV?

- 1. Virtualization:** Use network resource without worrying about where it is physically located, how much it is, how it is organized, etc.
- 2. Orchestration:** Manage thousands of devices
- 3. Programmable:** Should be able to change behavior on the fly.
- 4. Dynamic Scaling:** Should be able to change size, quantity, as a F(load)
- 5. Automation:** Let machines / software do humans' work
- 6. Visibility:** Monitor resources, connectivity
- 7. Performance:** Optimize network device utilization
- 8. Multi-tenancy:** Slice the network for different customers (as-a-Service)
- 9. Service Integration:** Let network management play nice with OSS/BSS
- 10. Openness:** Full choice of modular plug-ins

These are exactly the same reasons why we need/want SDN.

Source: Adapted from Raj Jain

Network Function Virtualization Advanced Computer Networks

To wrap up, let us try to see why we need NFV and try to understand how NFV is different from SDN, that we learned a week earlier and also try to see what are the key research areas and challenges that are still to be worked out and what is the way ahead with NFV. Foremost, if we look at what this NFV relies on, it is about virtualization, and this virtualization is basically to abstract out the physical hardware and enable the network appliances, which were dedicated hardware, to be now built as software entities.

And this means that we can build these network resources as software without really worrying about where it is physically located, how many of the instances we have to get, and how we plumb or connect these interfaces in a physical way. And also we have seen this helps enable a lot of savings on capital expenses, which otherwise you would have to shell out for buying these network appliances.

Further, it also helps enable to orchestrate many of the instances of these software functions, like if I can spawn hundreds of processes that is equivalent to creating hundreds of network

appliances. I can create hundreds of containers or hundreds of VMs it is simple for a network operator. Hence the orchestration, in essence, becomes a lot more simpler, and the time to market any of the utilities would be much more shorter.

Also, on the context when we looked this orchestration, if you have thousands of devices, it is good, but how do you really manage them? So, cataloging, keeping track of monitoring, and trying to do the best of the orchestration the automation becomes, in essence, the need. But because it is virtualized because it is software, we can think of programs that can also help us do this.

Next, because we have virtualized and we are trying to build these as the entities that are software that readily means that we have an infrastructure where the network appliances themselves are programmable, and by programmable what really gives us the flexibility is to adapt and change on the fly.

Like if we see there is a change that we want to try out, we can change the code, compile, run, and then we have the desired features to test out. And that is where programmability has been the means to say that yes we have softwareized now these network appliances and have decoupled all the dependencies that we had with hardware manufacturers or the OEMs and wait for them to deliver the required or meet the requirements and deliver us the required product. All of this is now decoupled.

Second, because they are programmable and they are software instances, we can scale them as we want. That means we can meet to the dynamic characteristics of the network, like traffic dynamics, whenever they are changing, we can adapt our instances of network functions as we want, and also this dynamic scaling can be signed in two specific aspects. One, we could scale out the instances and scale down or scale in the instances or lower the number of instances as we go.

We could even change because these are what think of virtual machines where I can even change the CPU, number of CPU, number of memory, what is the size of memory that I would want. All these characteristics without really trying to do anything on the hardware. Thus this dynamic scaling supports both horizontal as well as vertical scaling for us.

Third, now, because they are programmable and instances that we can control, we can automate the things much more readily and easily, and we do not need as much of a human intervention which is necessary for network operators or telecom operators to look at and manage specific devices but now we can offload the task to the machines in the form of newer programs that we can develop to automate the systems overall.

Further, whenever we have the hardware and they are deployed at one place, the visibility is possible only when we monitor those resources specifically and have the connectivity to them all the time. But now like as SDN said, there is a centralization, there is a central controller that can provide the entire information about the network topology. We can think of having these instances are also being managed somewhat by a similar management and orchestration framework, which can provide us the entire visibility of what is the status of each of the network functions.

Although we may not need to go and configure at each of the physical instances where they are deployed because they are virtual, like an open stack framework, we can check and see what is the status of each of the machines and likewise, like if you are using Kubernetes for the orchestration of containers we could also manage the status of each of the containers that will bring and it would really help us provide better visibility over what is happening.

And also, because now it is softwareized, the entire program the way the packets will be processed, the way the functions would act, all of this is under our visibility. If we have the open infrastructure and open source implementations of the network functions, unless where you had proprietary devices where only the specific OEMs would be able to look and see what the device is because it is a packed device where the firmware sign upgrades all of those would be managed completely by the OEMs but now because these are software, we can even do the upgrades anything on our side if we have open source implementations where we can change, adapt and update as we need.

Also, the performance aspects, when we see the requirements and we want to map, we may not have to shell out as much money for the hardware without the need. That means we can tailor the network instances, the softwareized instances to meet our performance requirements. This way,, we can actually optimize on the network device utilization.

Although, like I mentioned earlier, the performance when it comes to migrating from hardware to software, the customers to the commodity hardware, and virtual instances running on top of commodity hardware performance with NFV is a major challenge.

Also, if we look at it from a network and telecom operator perspective, what is more crucial? Like if you have a data center, I have a cloud. I want to manage multiple of the customers or multiple tenants, and all of those I do not have to procure newer hardware for each of my tenants. I could reuse the same hardware infrastructure for multiple tenants, and that is what we also discussed about multi-tenancy, which is now because of the virtualized infrastructure it becomes a lot more easier, and the physical infrastructures can be upgraded as needed. While we can support multiple tenants on the same physical infrastructure and this also means that we are able to consolidate and better utilize the resources.

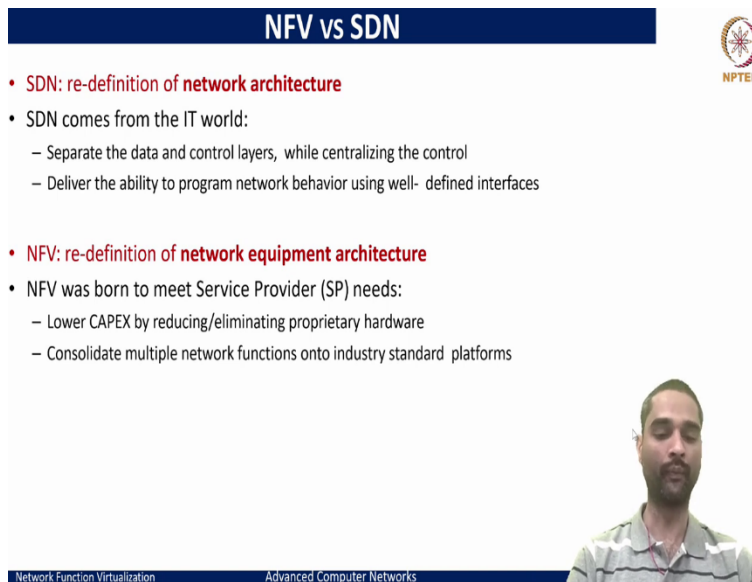
And overall, even the service integration makes it a lot more simpler because now we can have the operations support systems and business support systems, which would be decoupled and manage the network virtual infrastructures lot more easily.

In essence, the most prominent factor, which is the key here, is the openness wherein because we have made everything softwarized, it allows any researcher, any startup to come up create newer network functions, and innovate much more readily using the frameworks that have been created and for a network operator or a telecom operator perspective you are no more tied or hooked to a particular OEM vendor. You can mix and match any of the modules from different software vendors.

Thus this allows for a full choice on both the network operator side, on the developer side the implementers, and researcher side and openness fosters innovation, and this is how the NFV really changes radically the way we can think of the network infrastructure or build the future network infrastructure.

And if we argument and look back, the entire essence that we have captured here is the same as why we really wanted SDN. But the context was just a bit different. Here it was primarily from the network infrastructure point of view, and there it was basically in terms of how to control the network elements themselves. So, there has been also a debate in terms of whether SDN and NFV are independent or whether there is any dependency on the two.

(Refer Slide Time: 8:59)



The slide features a dark blue header with the text "NFV vs SDN" in white. To the right of the header is the NPTEL logo, which consists of a circular emblem with a star-like pattern and the text "NPTEL" below it. The main content of the slide is a list of bullet points. The first bullet point is "SDN: re-definition of network architecture", followed by "SDN comes from the IT world:" with two sub-bullets: "Separate the data and control layers, while centralizing the control" and "Deliver the ability to program network behavior using well-defined interfaces". The second main bullet point is "NFV: re-definition of network equipment architecture", followed by "NFV was born to meet Service Provider (SP) needs:" with two sub-bullets: "Lower CAPEX by reducing/eliminating proprietary hardware" and "Consolidate multiple network functions onto industry standard platforms". At the bottom of the slide, there is a small video inset of a man with a beard and short hair, wearing a light-colored shirt, looking directly at the camera. Below the video inset, there is a dark blue bar with the text "Network Function Virtualization" on the left and "Advanced Computer Networks" on the right.

- **SDN: re-definition of network architecture**
- SDN comes from the IT world:
 - Separate the data and control layers, while centralizing the control
 - Deliver the ability to program network behavior using well-defined interfaces
- **NFV: re-definition of network equipment architecture**
- NFV was born to meet Service Provider (SP) needs:
 - Lower CAPEX by reducing/eliminating proprietary hardware
 - Consolidate multiple network functions onto industry standard platforms

So, to clarify that, let us try to relook or revisit NFV and SDN in terms of the core principles and what is their model of operation. When we see SDN, it is basically the redefinition of the network architecture itself. Like how the network was built in terms of how you want to control and modify the behavior of the networking elements.

So, that is where we spoke about the data and control plane apps distinction and abstractions and having to do with centralized controls in a minimal sense, and this is where we came up with this set of abstractions and interfaces northbound and southbound interfaces to control the network elements.

And the origin for the SDN was primarily from the IT world and how the networks could be easily be experimented with. Like we want to get the visibility, get the view, and change any of the network protocols, all of this in terms of not exactly saying that we will re-build a new set of topology, we could use the virtual topologies and use the virtual switches and build SDN on top of it. Because of the well-defined interfaces, we could use the open flow to control these physical or virtual switches and manage them a lot more easily.

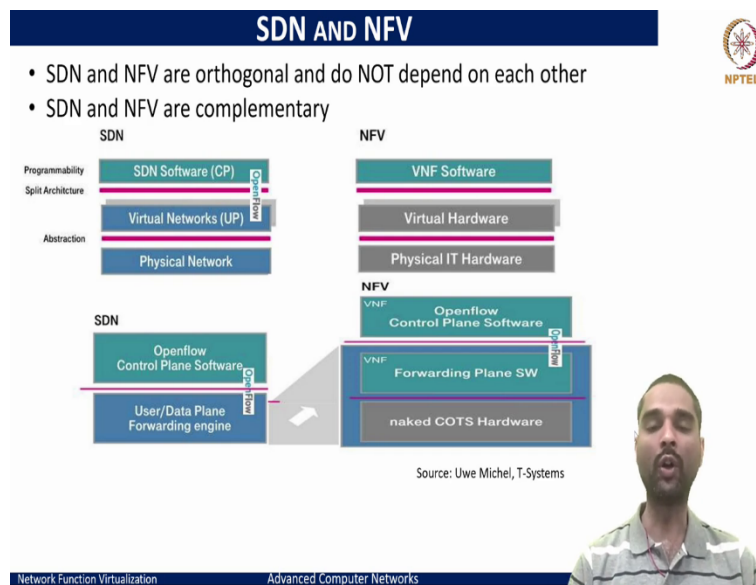
While with NFV, what we have seen is the redefinition of the network equipment architecture or network equipment themselves in terms of how this network equipment are built. Think of a router as a network equipment, switch as a network equipment. All of these need not be

proprietary devices, the NAT as a network equipment, firewall as a network equipment. None of these need to be built as a proprietary hardware.

So, we have now redefined that we can rebuild these as software functions, and this is NFV primarily came because of the proliferation of these middleboxes adding to lot of pain points for the telecommunication and network service providers requirements. And that is where NFV try to address those aspects.

And also, by product, what we have seen is also it enabled to greatly lower the CAPEX and essentially eliminate or minimize the dependence on the proprietary hardware and the consolidation of these onto a single infrastructure while leveraging what the IT world already use as cloud infrastructures to build this NFV on industry-standard platforms.

(Refer Slide Time: 11:22)



Further, if I have to put it in perspective what SDN provides is programmability, the split architecture of how the networks can be viewed with a control plane and data plane or the user plane where SDN software or the SDN controller would sit somewhere outside of the actual networking devices and the networking devices systems would facilitate basically the virtual networks data plane functionalities in the physical network.

So, that is how where we defined the right set of abstractions and split the architecture of the network elements. On the other side, if we see the NFV, we are now trying to say that the network function is now becoming a software which is basically a VNF software which is going

to be run on a virtual or a physical hardware so you have right set of abstractions that are being defined, and the programmability comes with respect to making these network functions as software.

Well, the abstractions come in terms of making this network function to run on a virtual hardware or physical hardware the way we would need and abstractions to build the virtual hardware on top of the physical hardware, which were, in essence, what the cloud virtualization that leverage.

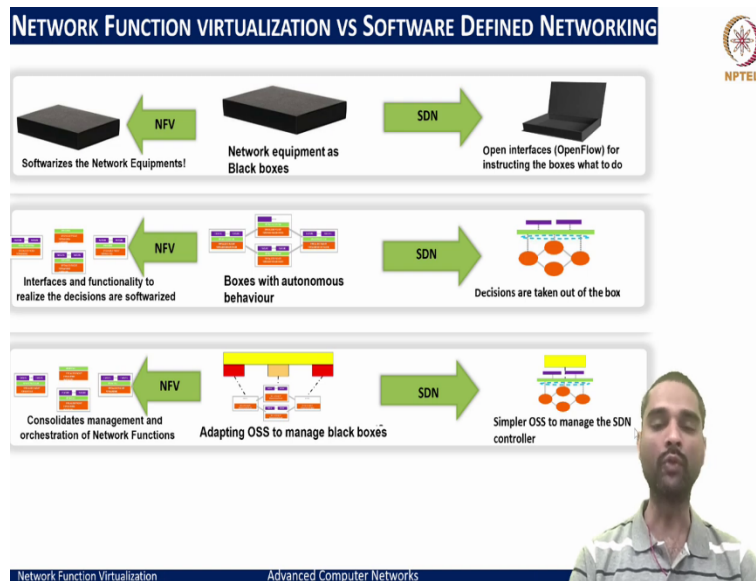
And in another essence, the three key things that came is the OpenFlow control plane software, that is SDN controllers or the network operating systems that would interact using the OpenFlow onto the user data plane forwarding engine. That is basically the networks and network switches and routers to configure the rules.

Now if we argument this to the NFV context. We can now think of the OpenFlow control plane software itself as a virtual network function. That is a virtualized network function that we can run and this could be basically our the network operating system that is being run as a VNF. This could be a single instance, multi-instance, a distributed system that can be developed as a virtual network function.

Second with the openflow abstractions that we now see, the forwarding element themselves like what we call as the switches and routers these are also the hardware that, in fact, in the SDN that we would want to talk to on the data plane. Now these instances themselves could be virtualized so you can think of both SDN's control plane and data plane elements as having being realized has virtualized network functions and when we think of a vSwitch that we built within our machines to enable communication across multiple VMs, the forwarding plane is a virtualized network function that we can really build as what are the means not necessarily use the same entire vSwitch for the communication but customize that because it is software now and adapt to our requirements. And all of this would eventually run on the commodity off-the-shelf hardware making full advantage and leveraging the means to build the network infrastructure at our will. And this is how we can say that SDN and NFV are both orthogonal and there is no dependence on each other.

While SDN and NFV, now you can see that NFV is complementary to SDN, and if you remember in the earlier talk, when I said for NFV, we have a management and orchestration framework, and for that SDN controller, the OpenFlow control plane software really augments to the way we would want to manage these network functions because we are now able to control and configure each of the devices through software. So, in essence, the two SDN and NFV can be really seen to be complementary to each other.

(Refer Slide Time: 14:56)



Further, let me put it in very simple perspective of trying to understand how we can think of SDN and NFV at different planes. At a very basic thing what it started with NFV, we had a network equipment that switches and routers as black boxes because they embedded both the control and data plane. What SDN tried to do is decouple the control and data plane and push for open interfaces, especially in OpenFlow for instructing these boxes, which were earlier black boxes, to now a means to configure these boxes the forwarding rules and set up how these devices should behave.

To this, now if we look at NFV, what it has tried to do is, do not think of network equipment as something that is hardware, but we can think of this as a software entity in itself. So, now no more that a switch is a physical switch but it is also a virtual switch and that is where the NFV plugs in. So, now you can see that SDN on one end can operate on both the physical switches as well as the virtualized switches.

So, both are again complementary here and both augment rules and without having the SDN, you could still softwarize and build the network equipment as softwarized network equipment. So, there is no dependency either in either of them.

Taking it to the second level and see how these devices behave, like when we abstracted the control plane, our intention was to make sure that we can centralize the way the decisions are made and push onto these networking devices. So that decisions are now taken out of the box on a centralized controller and all the devices basically would follow what the forwarding rules are being said and operate at a user plane or a data plane to just do processing on the packets as being instructed by the SDN controller.

Now in this space if we think of this decoupling that we have got and on the NFV side, how this augments to SDN is to really create the interfaces and functionalities to realize the decisions that we want to make because on the SDN controller you are really trying to run some functions and these functions that you want to build can be thought of as a software entities that can be built like we discussed about Dijkstra's routing.

That is one of the network function instance that could be running on top of the SDN control plane which dictates exactly what functionality needs to be done and enforce that decision through the SDN controller again.

So, again we can see that the functionality what we want to bring is where NFV augments to the SDN but and the last in terms of the essence that we can see what SDN through centralized control, what we really gained was a very simple management at one place for a network operator to look update the aspects and that was the SDN controller through which we could control the entire topology. And that means operations system support for managing those black boxes was made lot simpler.

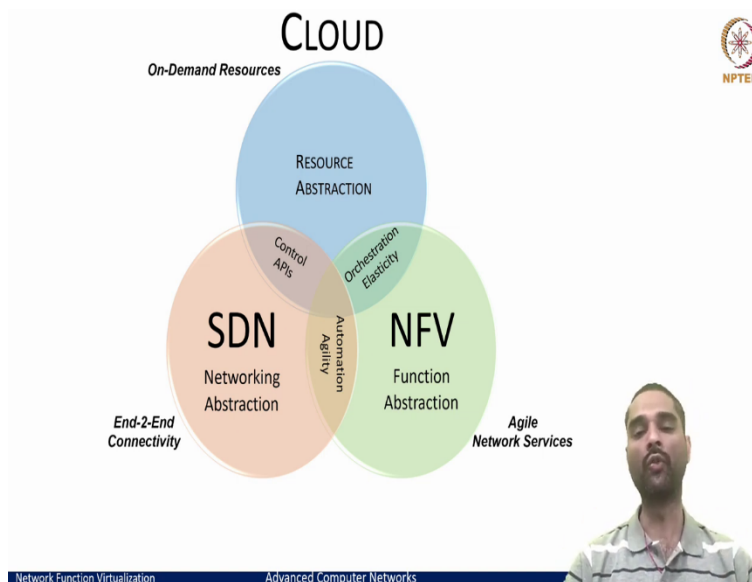
And now, with NFV, when we think of this, if you see that these instances could as well be consolidated into a smaller set of instances and orchestrated as network functions, the management can as well be done from the NFV's manual framework wherein you would use controller as a NFV and plump what are the requirements that you need to manage these devices.

So, again here we can see that because we are able to consolidate the management, we are, in essence, aiding and helping SDN to better orchestrate and build the rules, and in other way SDN

is augmenting for helping a centralized way to control and manage these devices and how to really provision network devices what to monitor all of these can work together between SDN and NFV.

So, I hope we have tried to clarify why these two are complementary and how these two can work on the same principles for different aspects and build a better softwarized networks.

(Refer Slide Time: 19:01)



To sum up, what I really want you to also remember is, we will address about the cloud as data centers in the later part. But the core principles of cloud is infinite resources or on-demand resources. And the way the cloud APIs are built is to provide you complete resource abstraction wherein you are completely decoupled with what is the actual physical infrastructure that a cloud provider is operating on.

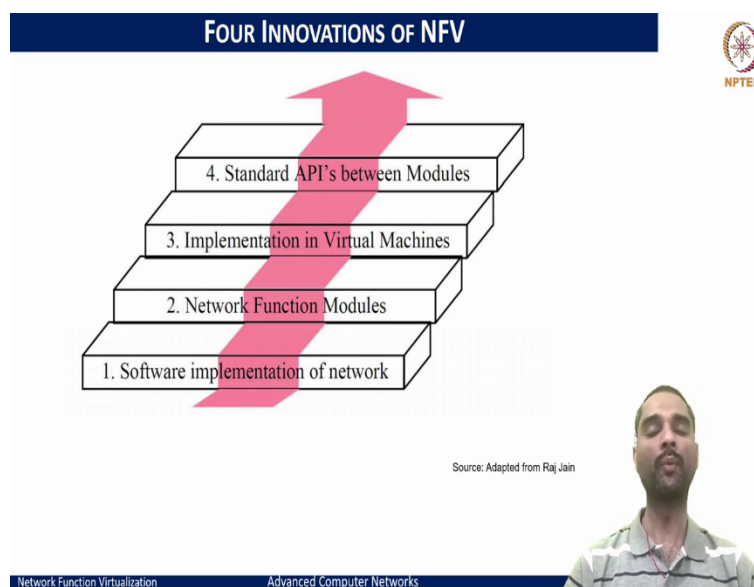
You are only concerned with your instances of paid infrastructure that you virtually build or be it the software or platform you could work on the clouds, and that is the benefit that cloud brought. To that, when we augment the SDN, SDN provides what we call end-to-end connectivity. That is end-to-end visibility of what goes on and provides the right abstractions for the network in terms of how the network control needs to be set, and that is where SDN can be summed up as a network abstraction that facilitates the right control APIs to manage the underlying infrastructure of virtual or physical devices within the cloud.

And NFV, on the other end, when we argue it requires a cloud-like infrastructure, and that is where you can basically do the orchestration of the provisioning and orchestration of multiple devices with providing the flexibility of elasticity as on-demand resources. But what NFV, in doing so, does is basically the functional abstraction because now we have built the network appliances, which were hardware as network functions which are now software that can be pulled in and scaled elastically as we want.

So, what NFV really tries to provide is having agile network services because they are very flexible, they are softwarized, we can spawn and recreate change, tweak, and adapt the requirements based on the requirements that we have very easily, and that is where NFV adds on to the agile network services. And how SDN and NFV really interplay is basically with respect to providing agility for the underlying network infrastructure and the means for automation.

Like when we want to control the NF infrastructure frameworks, we could leverage the SDN control to dictate how the devices should be, and when we want the NFV to basically manage, we can think of the control and data plane elements themselves that SDN wants to work with as the network functions. And this is how the three key pieces of cloud SDN, and NFV come together in trying to make us provide better-softwarized networks.

(Refer Slide Time: 21:38)



And to finally, remember about the key aspects of NFV; it is basically one that provides a software implementation of the network. That is, the network appliances are now software, and

what this also means that I can really break the network appliance into multiple network functions and use them as and how I want.

And this also means that if there is a firewall that is maintaining a state that is processing the packets, that is also doing a NAT kind of functionality. You could basically decouple all of these functions and realize these different functions that I want to scale. Especially when you see that you have a functionality being a bottleneck either due to the I/O or due to the compute, you could scale only such instances rather than scaling all other instances, and that is where this network function modules really helps us treat these microservices aspects into these network functions and scale them as and how we need.

And third, is basically implementation in virtual machines; what that means is these network functions could be implemented on a virtual infrastructure, and this way, we really leverage all the benefits of what we saw on the cloud or IT industry with the deployment of virtual machines. Abstraction for infinite resources, of course, we can get that if we are able to scale the resources and have sufficient underlying hardware to provision them on will.


And the fourth is the standardized APIs between the modules that we looked up earlier about what the HC's NFV reference framework and architecture try to pull about how we can enable any of the independent software vendors to develop the network functions leverage on like the open stack or Kubernetes kind of a framework for managing the underlying infrastructure and presenter virtual infrastructure where we could run virtual machines or containers as we see fit and also make sure that they are able to operate, interoperate amongst them through the defined set of standard APIs.


And the road ahead is with respect to how this NFV is shaping the 5G and 6G networks infrastructure, the Edge infrastructures, and HC; if you visit the HC NFV working group, there are specifications and documents about how the NFV infrastructure or what are the specifications to meet out the 5G requirements, to meet out the Edge capabilities that the community is currently talking about and it is playing the road ahead.

(Refer Slide Time: 24:09)

NFV AND SFC RESEARCH AREAS

<p>NFV and SFC architecture</p> <p>Abstractions and Performance Abstractions for carrier-grade networks and services Performance studies (optimisation, scheduling, portability, reliability)</p> <p>Protocols Protocols for Service Function Chaining Protocols for Infrastructure Management and control automation</p> <p>Evolving Architectures New requirements on the NFV Infrastructure for supporting new types of VNFs NFV Infrastructure federation New network topologies and architectures Tools and simulation platforms</p>	<p>NFV and SFC Systems and Applications</p> <p>NFV Management and Orchestration Service chaining algorithms & NFV orchestration algorithms Orchestration: NOT a single point of failure Load balancing, Resource scaling Monitoring, synchronisation and trigger mechanisms in the event of failure of NFs</p> <p>Resiliency Chained resilience plans and Service Continuity SLA minimum insurance Correlated failures in NFV-FG NF state management and chain-wide consistency</p> <p>Security Secure separation and management of NF entities NFVI shared resources & Isolation of VNF sets User privilege resources access (APIs)</p>
---	---





Network Function Virtualization
Advanced Computer Networks

Let us try to look at the NFV and Service Function Chaining, in essence, put together as network services, what are the core research areas. The first and foremost like, when we think of NFV or SDN, it is again abstractions that we are trying to build, and with abstractions, also we want to ensure that the performance criteria are not affected. So, abstractions and ensuring performance-oriented abstractions is a key aspect. So, the abstractions for carrier grade networks and services and which are basically performance-oriented, so that you want to minimize the overheads of presenting the abstractions themselves. That is one of the areas, the other prominently being just the performance of the network functions themselves on a given infrastructure. How do you optimize, how do you schedule, load balance, and ensure that these network functions are reliable, becomes another critical area of research, and if we see in the last decade, many of the publications have come out that tried to address these abstractions performance and also the protocols aspect when we say Service Function Chaining, how do we ensure that the packets go back and forth with less of overheads? How do you steer the packets towards the desired network functions, which are the intermediaries that are being built transparently?

So, there were several of the protocols like the prominent one being the Network Service Header NSH that was put forth by Cisco, and there were also many other research works like steering, flow tags, and many of the other works that followed to say what is the optimal way to use and realize this function chaining.

And also the protocols for infrastructure management and control automation themselves could be the key aspects to see how we are able to build the protocols that are quick to react, that provide lower overhead, and so on. And overall, this NFV as an architecture in itself presents a lot of new aspects to think of like when we have the clouds, and we have public and private clouds, they have to interoperate, and likewise, if we have a federated cloud, the same could work good for NFVs to have an infrastructure that is federated.

New kinds of network topologies and architectures that we can really think of and for all of this as a researcher or as recognitions, if you want to exercise and work out in a much simpler manner where we may not have access to any of the physical infrastructure or test beds, we would also need the tools and simulation platforms to try out and see how things will work and like on cloud you have the simulation frameworks you would try to want to see how the things would be on our individual machines using some of the simulations or at least on a local very small test bed to try out different things.

On the other side, if we see it in NFV and SFC systems and applications and well on the application side, there is no limit to innovation, and we can have lots of new novelties, especially with the IML coming in; how do you want to do real-time inferences on specific things, deploy them at edges as edge network functions. What kind of monitoring we would want to do to ensure that the SLA requirements, wherever they are violated we are able to predict, course correct, and build the right actions within the edge and telecommunication networks point of view and also orchestration like it is now no more a single point of failure because we can have these virtualized network functions themselves trying to be built as an orchestrators, how they would coordinate communicate and build this framework are all some of the interesting challenges.

Further, when we look at resiliency, we spoke about what are the carrier grade requirements in terms of downtime, in terms of time to recover from failure. So, how do we meet all of them with NFV or the NFV infrastructure that we discussed above? So, also, for the network services, because now you have a number of network functions that are chained to provide you these services, it could be that not cut the hardware frame, but one of the network function instances failed.

That means the chain in the chain multiple other instances are still active and working while one of them has failed, and it is actually affecting end-to-end service. So, how can we overcome this kind of service chain breaks that would happen due to network functions going down or a link or virtual link or physical link that is connecting having some issues?

So, a chain-wide resiliency plan and also this is where consistency also plays a major role because once the packets are processed, the service is not complete until it is processed by all the network functions. But if one in two of the functions has been processed while the remaining are yet to be processed, it is no more having the atomic operations.

So, we want to build the operations of a network service as atomic, starting from the first NF in the chain to the last NF. So, how do we want to build such a framework without adding much of overheads? That is another core research area in terms of NFV. And whenever there are failures, you want to see how is these relations that are being built for the failures, how do you correlate them, and see how do you want to build the failure resistance network functions in this aspect.

And also tied with this is the state that we are trying to talk about because all network functions have state; very few are stateless, but the majority of the network functions are stateful. So, how do you manage the state and ensure the consistency for operations are all the key areas of aspects to be addressed in NFV resiliency, and the other aspect is always often the security which needs the right abstractions and right separations so that you are able to manage the network functional entities without leaking or any of the entities.

And like I said earlier also these network functions because now they are going to be coming from multiple independent software vendors, you want to ensure the isolation characteristics for these devices, and also you want to have a means to trust which network functions you would want to build, what functions that you can really deploy on your frameworks.

And also once you deploy these functions likewise when you deploy an app on your mobile phone, a user needs to govern what are the access privileges for those network functions. The same way user needs to dictate and control what are the access privileges for the network infrastructure that we have on which these virtualized network functions are going to be run. So, that means you need better and newer APIs to facilitate this kind of privilege control access authorization and access controls for these NFVs.

So, overall we have looked at the NFV and said how NFV helps towards building the better network infrastructure and why it is a key aspect in network softwarization and also discussed about the Service Function Chains. Later off we will start to look at what we would try to see as programmable networks, which is an extension to the SDN world in terms of seeing how programmability can be done to address what we call as not just the traditional network stack but also if I want to customize and build my own network stack, what do I do? That means I need further programmability to build such things, and we will start to look at that in the next week.