Cisco Cloud Unfiltered Podcast Series, Episode 14: Vikram Hosakote

In this episode, OpenStack Community member and Cisco Senior Software Engineer Vikram Hosakote explains what's going on with OpenDaylight right now, why Kubernetes is so popular, what's up with Cisco Metacloud, and the difference between what enterprises and service providers want and need in their cloud solutions.

Niki Acosta:	Hello, hello, hello. This is Cloud Unfiltered. I am your hostess Niki Acosta.
Val Benincosa:	Hey, I'm Vallard. It's good to be with you again, Niki.
Niki Acosta:	We have an awesome guest with us today. Somebody who's been at Cisco for a minute, knows a lot about OpenStack, knows a lot about kind of what's happening in the community with containers. It is Vikram Hokasote. Say hi, Vikram.
Vikram Hosakote:	Hey, Niki. Hey, Vallard, how are you? Thanks for having me on the podcast. I'm Vikram Hosakote. I work in Cisco's Metacloud team, mainly working on OpenStack and deploying and developing cloud products for Kolla customers. I am also working in containers. I'm a core-reviewer of the OpenStack Kolla and Kolla-Ansible projects string.
Niki Acosta:	Cool.
Val Benincosa:	Very cool.
Niki Acosta:	We usually start things and, by the way, thank you for clarifying just before we started on how to pronounce your last name because I have been saying it wrong.
Vikram Hosakote:	Yeah, Hosakote.
Niki Acosta:	Thanks for clarifying. We're really excited to have you on today and we typically start these things by asking you about how you got into tech, what were you like as a kid, how did you find yourself on this path that you're now on?
Vikram Hosakote:	Yeah. Let's see, I was always a curious kid, right? I remember I broke open TV remote control, I used to break open my remote control cars, I used to break open all the electronic gadgets at home. I remember doing that. But as far as computer science

itself is concerned, I want to say maybe when I was like, yeah, like 11 or 12, I got introduced to a computer. It wasn't even a Windows, it was some type of X86 Xbox. There was Basic on it and one of my teachers at school was working on it. That's when I got introduced to computer programming.

Then also, we've got a lot of books about programming, things like Fortran. I also saw COBOL people working on it at my school. It all started back when I was like 12, 13, but not very seriously. Then, end of high school-

Val Benincosa: Wait, wait, wait. Quick question, can you still program in Fortran and COBOL?

Vikram Hosakote: No, no. I started in ... I remember when I got into Fortran there was an earlier version, I think it was Fortran 77. I never understood that. I think I didn't understand the syntax. I can probably brush up my Fortran 95 skills. It was one of my first program languages. COBOL, because they were doing a lot of mainframe and batch processing work at my school.

Val Benincosa: Yeah.

Vikram Hosakote: I see people working on COBOL. The thing I liked with COBOL is the whole divisions. Right? There's an identification division, there's a data division, there's a procedure division. There's lots of divisions.

Val Benincosa: I just like that you're defending and talking about how radical wall is.

Vikram Hosakote: Yeah, yeah.

Val Benincosa: Did he grow up in the Boston area then?

Vikram Hosakote: No, no. I grew up in India actually.

Val Benincosa: Okay.

Vikram Hosakote: I did my schooling and my high school back in India where my school had a lot of data exchange, they were working on mainframe, batch processing, large transaction processing, mainly on Basic, Fortran, COBOL, and a lot of books, and professors working on it. End of high school, I remember I was part of a supports team that kind of did support work for all the computers in my school. There were no laptops back then. We used to do any sort of help as far as computers are concern like disk or drivers.

I remember I got introduced to Linux as well. No CentOS or Ubuntu, this was all in Solaris or maybe BSD, I think FreeBSD, NetBSD, on those days. I remember we used to help people load drivers for printers or the network interface cards. That's how I got introduced to tech.

After that, I got my degree science undergrad and then did my masters in computer science. Then, pretty much that's it. I actually started PhD in computer science and dropped it after seven months, and I joined a start-up because everybody in my research group did that. I also wasn't, I think, I wasn't focused to take it from computer science school. Then, yeah, I came to Cisco, saw the open source world, started loving it and OpenStack.

Niki Acosta: What is it like growing up in India? Because I think there was like this perception that you kind of have a destiny to either go into computer science or go into healthcare, like there's specific fields that if you're going to college, what was your path like? Is that perception off?

Vikram Hosakote: No, no. You're very right. My dad is an electrical engineer, all my uncles are engineers, many of my cousins are engineers and PhD. I think it's maybe a society thing or maybe when my parents grew up the jobs that kind of earned good salaries were like engineer and doctor and lawyer. Right?

Val Benincosa: Right.

Vikram Hosakote: For some reason, either it was the political system or the bureaucratic system, lawyer was, at least in my family, my dad was like, all lawyers are fraud and they take a lot of bribe and stuff like that, but in my family, you gotta be somebody who gets a degree that gets you a good salary at end of the day. Right? I mean, you can go into manicure and arts and all of that, but I don't want you to be asking for more my money or come to my basement [inaudible 00:06:20].

So it's kind of like that attitude, and then, yeah. That's why most of the people are either engineer or doctor, or lawyer, mainly because everybody's doing it. Right? They're seeing good results. They're doing great jobs, good salaries. I remember people who studied engineering and doctors, they had mortgage and would buy a car in their late 20s. When my dad's generation did a PhD in other branches and they weren't so financially settled. Settled like in the society's eyes.

It's kind of like a go-with-the-flow thing, but if you look at the big picture, it helped me because it put me into tech. I loved tech, so it's worked out pretty well for me.

Val Benincosa: Yeah. It's good. It's good for you. Let's talk about the tech, man. What's exciting you these days, in cloud and everything?

Vikram Hosakote: Sure. Let's see, I got involved in open ... Actually, yeah, my involvement with open source was my first step into cloud. That was back in 2011. My ... not my previous, my two works before my current organization. We were doing a lot of VMware projects, like we had VMware products to deploy virtual machines. I remember back in 2011 we started using or moving from physical metal servers to virtual machines. Right? That was when I got involved with open source because we were using CentOS virtual machines a lot. CentOS use libvirt, right? For virtualization, libvirt is the API layer, the tops down to the virtualization, like KVM.

Libvirt was broken in 2011. They asked me to find a fix and I started looking into the code, it was C code. Then I told them, "I think I kind of found a fix which might work." Then they said, "You don't have to find a fix and implement it." They said, "When we mean find a fix, go to the open source community and ask them to fix." That's when it kind of got flash bulb, oh, okay, that's how open source works, so somebody else does a lot of work for us.

Val Benincosa: You had actually gone into the code and you had figured out exactly what's wrong?

Vikram Hosakote: I had, yeah. I had booked my own libvirt works with whatever fix that I was about to cast. Then one of my architects, he said, "It's an open source project. You don't have to even worry about fixing or finding the code. You can go to the community to talk with developers." That's when I sent an email to libvirt developers, told them the problem. They sent me a patch immediately and it worked beautifully.

That's when I started liking open source, just the pure anonymity of open source. Right? You don't know who is working on it. You don't know where they're from. You don't know their gender. Like in the IRC, for example, when you're chatting, you wouldn't know where they're from, what they're working on, all you know is they like what you like and they're working on what I'm working on.

Val Benincosa: Yeah.

Vikram Hosakote: That's how I got involved in open source back in 2011. My first moments to cloud was in the OpenDaylight project. Yeah, that was back in 2013 where Cisco built an SDN controller. It's called the POX controller where Cisco had this idea to implement an SDN design to kind of separate the data plane and the control plane, kind of opposite the traditional Cisco model where you have a box the dusted plane, control plane, and data plane, everything. They thought once the data plane is all figured out and burnt into the hardware the control plane was pretty idle. It wasn't as busy as data plane.

Then Cisco went on to separate the two things and have a separate box, the disc control plane, which is the network intelligence part, and then the data plane, which is, as they say the dumber part, like just kind of like the forwarding engine. That's when I-

Val Benincosa: What's going on to OpenDaylight project right now?

Vikram Hosakote: Yeah. It's very active. I was involved in the Hydrogen release. That was the first release. OpenDaylight follows the names of the elements in the periodic table. I think they are Boron now. They started with Hydrogen, I remember in 2013. That's when OpenDaylight was open source in the first place. Yeah, yeah. Then I think there was Helium. Now it's Boron. There's Carbon. I think, yeah, I think Carbon is next, but I don't know, but it's not as easy as OpenStack.

Niki Acosta: It's H He Li Be B C N O F Ne, is the first two rows of the periodic elements. I had it on-

Vikram Hosakote: Oh, yes, right.

Val Benincosa: I did remember that.

Niki Acosta: H He Li Be B C N O F Ne. That's how I remember the first two rows. We had a teacher that made us remember those. Thank you, Mr. Hall, appreciate it.

Val Benincosa: Are people really using OpenDaylight? I hear Cisco talking about it, but I don't know that I've heard, and it could be that I'm not looking in that space, but I'm just curious, are you seeing it, why do people want to use it?

Vikram Hosakote: Yeah. OpenDaylight is actually pretty active, especially the latest release, the Boron release. Basically it's kind of different than OpenStack. Right? OpenStack is for cloud, right? It's a VM as a service.

Val Benincosa: Yeah.

Vikram Hosakote: If a customer wants to replace their metal servers with a bunch of virtual machines and network, then they go to OpenStack. But OpenDaylight is completely different. It's more for service chaining, for example, or ideally use case would be like ... There are still virtual machines in an OpenDaylight environment, but they're not as independent and standalone like an OpenStack environment.

An OpenDaylight would have a chain of machines one after the other, like a virtual rack or a virtual switch that works with firewall, works with load balancer, works with proxy. They call it the ...

Val Benincosa: Yeah, service chaining.

Vikram Hosakote: Yeah. The service chaining or they call it a channel sometimes. They're all in chains in the OpenDaylight world. When data comes into OpenDaylight it goes into service after service in service chain.

Val Benincosa: Are there vendors around it?

Vikram Hosakote: Yeah, Cisco. When I was working OpenDaylight, Cisco did the POX controller. Nicira Networks I think did the NOX controller. They were the initial SDN controllers for OpenDaylight. But today OpenDaylight talks more than just OpenFlow, it talks SNMP, it talks REST API, it talks REST CONF. It even talks Neutron and integrates into OpenStack. That's actually one of my talks and workshops at Cisco live in Vegas last month. We showed how OpenStack, which is a cloud use case, can be integrated with OpenDaylight, which is an NFV use case, and both can be deployed. NFV customer who wants a cloud can use the integration and get OpenStack or an enterprise customer who wants a service chain can also do the same and get the other part of integration. Yes, it's pretty active. It's not as ... I mean, the developer it's not as big as an OpenStack.

Val Benincosa: Of course not. Yeah.

Vikram Hosakote: NFV use cases are not as much, or they're as widely seen as OpenStack. Telecom uses and some finance uses I think.

Val Benincosa: Yeah.

Vikram Hosakote: But traditional data center enterprise, retail, all they want is just OpenStack like virtual machines, firewall, and private cloud or maybe public cloud.

Val Benincosa: Yeah. If you can tell us what you're working on now, like what's exciting that you're working on now and what's the future of it, and why is it exciting to you?

Vikram Hosakote: Sure. I'm currently working on two projects. I am in the Lunar team in Metacloud. That's the networking project in OpenStack. We work on Neutron. We are right now upgrading Neutron to Ocata.

Val Benincosa: Wait, hold on. What's the Lunar?

Vikram Hosakote: Lunar is a team name.

Val Benincosa: Team name, okay. That's nothing to do with any OpenStack projects.

Vikram Hosakote: No. Team names at Metacloud are, I think they're based in galaxies or stars, kind of related to cloud, so there's sort of the lunar there, the galaxy. Lunar team does networking in the Metacloud, it works in Neutron. I am also working in the site reliability engineering team helping ops team work on production issues, and of course working on Kolla and Kolla-Ansible upstream. I'm a core reviewer of the Kolla projects so I hang out in the IRC channel and work with the Kolla developers.

Niki Acosta: Tell us what Kolla is.

Vikram Hosakote: Sure. Kolla was actually started by Cisco, or maybe it was upstream by Cisco. Kolla is a project that containerizes OpenStack. It was started, I think September, October 2014.

Niki Acosta: What does that mean to containerized OpenStack?

Vikram Hosakote: Yeah. Kolla means glue in Greek. It's kind of like different, very different than the traditional deployment model of a cloud. Kolla, what Kolla does is, it takes an OpenStack service, puts it in a docker container and then uses Ansible or Kubernetes to deploy in the data center. There are multiple advantages to this. Right?

Your question was a great question. Why do we want to containerize anything? OpenStack is pretty complicated to deploy. There are bunch of services, lot of big files, a lot of defaults, nondefaults, a lot of tuning, a lot of knobs, and then it's a fast-flowing project. Right? What works today may not work with the newer version of the same project tomorrow. Kolla or containerization makes it easier to get back to a golden state. That's very important in cloud format. We clearly saw it when we worked on Kolla. Because if something works it's not so easy to take a snapshot of the entire cloud and have a golden copy. Right? Kolla makes it possible so everything is baked inside something called as a container.

In the shipping world, and kind of an analogy, is like the shipping world where if we order something from Amazon.com it comes in a container or in the box. It comes on the ship from wherever it's shipped from, right? When we receive it we would sure that everything in it works out of box. If something doesn't work it's my mistake as a user. If the container works for me and if it doesn't work with somebody else, it's mostly because it's user's mistake.

That's why container is famous because it makes developers point fingers at others when something breaks and no other technology made that possible. Because all these days if something works in my system, if it doesn't work in your system, I have to come to your system look at the dependencies, look at what software, hardware you're running, and things like that, and fix it.

The container completely eliminates that problem, right? Because a container has everything, including the piece of code and it's unpacked with some config files needed for it to work with the software. When it works for me, it has to work for you. If it doesn't work for you, either your hardware is bad, it doesn't mean the containers at fault, or you're supplying software with some bad invalid configurations, which is again your mistake.

Val Benincosa: I didn't actually quite realize that. Kolla project isn't actually for users to run containers as part of their workflows, like something like Kubernetes provides, it's a way to stand up OpenStack and manage an OpenStack cluster.

Niki Acosta: OpenStack services, right? Yeah.

Val Benincosa: OpenStack services.

Niki Acosta: Yeah. Which I heard, and this might be a gross oversimplification, but the sort of reason intent for containerizing an OpenStack service is, for Metacloud, is that you have the ability to do in-place upgrades that are non-disruptive, which is huge. I mean, at that point, you've got all of your services, they're running a container, you can swap out a service in real-time and literally have less than a millisecond of downtime where the data plane is unable to either provision or decommission cloud resources.

Vikram Hosakote: Yeah. That's totally possible. What Niki brings up is a great point. That's all post-day 1 or day 2 activities. Kolla container is a great example to do day 2 or post-day 1 activities in a very easy, reliable way. For example, upgrades, very easy containers. Just pull in the new container and you can start them; reconfiguration, very easy, because the software itself doesn't change. Say you want to do some tuning, for example, we were talking how an enterprise cloud, for example, can be tuned to NFV, right? That's a

reconfiguration use case. It's like do the tweakings or tunings of configuration of the container. That's all again day 2. Patching, for example, there's a bug and we need to patch our software. It's not the mistake of the user in this case. It's very easy. Fix the software, build a new container, and ask the user to pull in a new container.

Yeah. Everything is easy post-day 2 because we just give them containers. So from an operations point of view the unit of deployment that's passed around or what ops will see and what they need to deploy is now containers. They don't need to worry about what packets to install, what dependencies, what config files, or what networking, what tweaks, and all that. Everything is baked into the container that we give them and all they need to supply is the runtime things when they start the container. Right? Which is called configuration. OpenStack has a million configuration servers when they combine-

Val Benincosa: Yeah.

Vikram Hosakote: You can mix and match and do a lot of stuff, tunings.

Val Benincosa: It's one of the drawbacks of flexibility, is the complexity can increase sometimes. What part of this project are you working on? You mentioned that you're an upstream ...?

Vikram Hosakote: Right now Kolla is moving to something like a Kubernetes model. The main reason Kubernetes is very, very famous is because, it's also containers in the first place, but users don't build anything. If a user wants Kubernetes they go to the website, they pull the containers that are already built in the Google, in the GCE. Right? Google compute engine data registry, container registry. They pull the containers and they start the containers and they use Kubernetes and they're good to go. But the Kolla, it's almost the same with an extra step before starting the containers, which is building the containers from scratch. That's been a pain for ops because building all the OpenStack services OpenStack has around close to 100 containers.

Val Benincosa: Yes.

Vikram Hosakote: Building them takes a good couple of five to six to eight hours. Sometimes building them fail because, if the register or something's down. Kolla is going to more where we give them built golden OpenStack containers that have to work out the box. They now can use Kolla and has OpenStack, has Kubernetes, like go to a registry, which is docker hub in Kolla's case, pull the OpenStack world in containers, supply the right config files, and start the containers, and boom, they get a working cloud.

That's way, way, way more easier than most of the ... or I want to say all of the server that work with Jinju. I started with Cisco OpenStack, and somehow I've run PAC stack, I've run OSAD, OpenStack Ansible deployment. Everything needs a lot of pre-configuration or being zeros to satellite talk.

Val Benincosa: Yeah.

Niki Acosta: It's trying to eliminate all that and just pull the containers and run it. It's like downloading the software and running it, right? We want Chrome or we want Slack, for example, in our lap. We don't want to build Slack from source. We just download it and we start it when we use it. That's how Kubernetes is and that's why it's very famous. Just pull it and you can run it. Nobody needs to build anything. It's for the developer or whoever develops containers eight to 10 hours a day. It's or her job to do it. As a user or ops, I don't even have to know what the container is.

Val Benincosa: If I'm going to deploy OpenStack with Kolla, then I would need to have just docker installed or I would need to get ...?

Vikram Hosakote: Yeah. You need some basic software installed, of course. You'd need docker installed, you need docker py, which is the Python client Ansible installed, some basic networking needed for Kolla. Any networking needed for OpenStack is taken care by Kolla installation. The good thing now is we are trying to automate that as well. If you give a very minimum stack, a pristine, I want to say like CentOS operating system, for example, with nothing installed, just some sort of IP to get into the box, Kolla is going to automate the pre-installed work as well, like deployed docker py Ansible, set up the network. Then pull the golden containers and start it. As a user, you need to just provide a basic VM or-

Val Benincosa: Just an operating system.

Niki Acosta: Basically something that used to take days and days and days and days and days, now you can do in minutes.

Vikram Hosakote: In minutes, right. Again, this is all in terms of, for developer environment, if you're looking for production cloud, the same containers can be used, but the networking part has to be set up. Because a production cloud has a lot of computes, controllers, storage nodes and they have like an upstream devices, like a couple of proper ax to tunes and all sort of bonding going on. Kolla doesn't do any of that. That's in fact outside of OpenStack.

In a production world, I'd say we still want to do a lot of day 0 work. That's going to be the case in any automation, right? Containers or not. The truck pulls in, the box are, I mean, the rackers are in box, they're racked in stack, cabling is done, the power is turned on, the operating system is loaded. That's what's done by the ops on day 0. Then day 1 is when Kolla start like pulled and gone.

Niki Acosta: You've spent some time working on private cloud products, both for service providers and then also for enterprise tech customers as well. What are some of the challenges or differences between what service providers want and what enterprises want when it comes to private cloud?

Vikram Hosakote: Yeah. It's great question actually. My work in OpenDaylight, which was mainly NFV and SP, was a good segue for me to move into SP and NFV use case for OpenStack. When I started working in OpenStack, my customers were all media customers, which is a service product NFV use case. The product that I was working on was called Mercury, which is today called virtual infrastructure manager or VIM. Basically, yes, there is a lot of difference for sure. SP, NFV telecom is a lot, lot different compared to an enterprise data center or retail, or something like that.

Yeah. As far as differences are concerned, it's just what the requirements are needed by the customer. I want to say a cloud, like enterprise data center use case, it's much simpler in the sense all they want is a bunch of virtual machines networked together, and that's it, they're good to go, and that the cloud is ready for use. They run their applications in the virtual machines and they're good to go.

But the SP, NFV was because I was working on was way, way more than creating couple of virtual machines, like they have this concept of service chaining, for example. They have this concept of channel. So media customers where I was working on, they wanted an entire channel in their data center. A channel is a set of virtual chain services one after the other, a kind of service chain, service chain channel and a couple different terms for the architecture. When the packet comes in into their data center they want each service to do something to the packet and then pass the packet to the next service in the chain.

For example, virtual rocker switching, which in firewall, proxy, load balancer, they can all be DMZ. They can all be connected as a chain and we can make, we can visualize a chain as something coming in into the chain, going through the chain, getting changed in the chain, or doing some sort of work in the chain, and then coming out of the chain.

That's very different compared to a data center enterprise case. It just works on chains and they're not much of chains. There's no dependency as to this has to run after that and that has to run before it. But in service chaining, like it's a chain, so if we have to run service 1, if that passes go to service 2. If that fails go to service 7. If service 7 passes, go to service eight. If that fails, go and retry in service 3. If service 3 passes, back it up in service 11 and then get out of the chain. So it's something like that.

Niki Acosta: Is that because of the nature of what service providers are doing, like in terms of just processing huge amounts of data all the time?

Vikram Hosakote: Yeah. Yeah, the provider service, right? In the SP world, something is being streamed. That streaming is very important, I forgot to mention that. In an SP cloud, we always see something being served to the client. For example, Netflix or I have IPTV in my house. Sling, for example. A Sling data center is definitely different than enterprise data center in the sense when I turn on my TV a bunch of services need to run in the data center of Sling or Netflix to provide my streaming on my TV. As long as I watch my TV, like five hours, 10 hours, or entire night those services better be working. It's not like a bunch of VMs running an application.

When I change my channel I may use a different service or the same service with the different streaming service, I mean, streaming usage, right? That's why Multicast was widely used in my work with the media companies where when we see a TV channel, which is broadcasting, for example, California. I am in Boston and say everybody is on the East Coast is watching the same TV channel, they don't have to replicate each packet all the way from California to all the East Coast. Okay? They don't have to replicate each packet a California to Boston, California to New York, and California to Atlanta. It doesn't make sense.

What they do is they probably have like, I don't know, Strabo, or somewhere middle of the country where they send just one stream and then their data center at the center of the country will have a Multicast service chain that knows how many receivers or how many people are watching this channel and that's going to blast out Multicast streams.

You see, this is not possible in the traditional non-service chain cloud, is it? I mean, how can we do any sort of providing a service at real-time with just bunch of virtual machines?

Niki Acosta: Yeah. It's not like a CDN where it's like permanent kind of cache there until the TTL expires. Right? I mean ...

Vikram Hosakote: Yeah. Yep. I mean, you could use CDA in SDN case. You can do content delivery network to deliver your content, but something is being provided or serviced all the time. If nobody watches Netflix, then Netflix might as well be an enterprise use case. Right? There's no service being provided. If nobody watches YouTube, then they won't need any service chaining. They'll just have a bunch of virtual machines and cloud enterprise. Right?

Niki Acosta: It's crazy how much we take all this for granted. Like when I'm browsing my phone late at night going through things, switching from one thing to another to another to another, and I click a button and it's there and it's there, it's there, I don't realize, I guess I don't

realize all of the hops or the complexity behind what it takes for that content to be previewed and then executed and delivered to me right now.

Vikram Hosakote: Yep. All these days, imagine that this was all being done by individual metal physical servers. So that's SP, I mean, NFV is taking off, right? The moment we can do all of these in a streaming and replicating packet from California to the other part of the country with just virtual machines, that's way, way easier for some suppliers. That's why they want to deploy all the services they've been providing all these days in a virtual environment. They call it a chain. It's a series of activities.

Niki Acosta: I'm not a networking expert, but when someone sends content, let's say, I'm in Austin, you're in Boston and, I don't know, we start this Google hops and we start talking. Is there an option for you to make sure that your data is getting to me on the easiest, shortest, cleanest route?

Vikram Hosakote: Well, SP, you're talking, you have to go to agree-

Niki Acosta: That was service provider's data, right?

Vikram Hosakote: Yeah. UDP, right? Unreliable. Yeah. That's very good point you bring up. Yes. If somebody's going to be servicing a lot of data to how many of their customers, there is an element of unreliability. Imagine doing all of these streaming in TCP with a lot of acknowledgement and sequence numbering. We would want to take up a lot of bandwidth and then, two, of course, we would get a confirmation that the other end is receiving the stream, but that's a lot of work for the hops in between.

The use case, you're talking about, I almost did something like that in the media space, with media customer, but they didn't worry about confirmation. They were blasting Multicast in DP streams. As a receiver of the data, for example, I'm in a blasting Multicast in DP streams, as a receiver, you are okay and kind of signed an SLA agreement that, okay, I can tolerate maybe 5% drop or maybe like 6% drop. A line or two, you can tolerate.

Niki Acosta: Yeah. There's times when my Google Hangout gets a little pixelated and the sound drops for a second, but then it catches up and it's good.

Vikram Hosakote: Yeah. It's not that once your network picks up you're not going to rewind and you're not going to automatically see what the video you missed. That's not how video works, right? If you've missed something, you've got a jitter, you lose it forever unless you've recorded it. That kind of how TV works, right? Seeing a live streaming like Superbowl and something goes wrong and the user don't worry. They agree that, they know that some sort of unreliable piece is okay.

That's a challenge in this viewer like how can you reduce this element of UDPness or unreliability in your service provider. Right? You can have fat links, you can have fast links all the way to user. I can get a one gig link to my router if I pay more to Comcast. But not everybody does that. Everybody has a regular house or household link coming off of some Comcast outlook. It's challenged to your speed that without using a big expensive fat link, how can I reduce the jitter of that streaming?

Niki Acosta: Yeah. Especially when you don't know what your user is paying for. Right?

Vikram Hosakote: Yeah. Yeah. That's a good point. Yeah. You don't know what user is paying for, you don't know what you're seeing, you don't know if it's live or not. I mean, if it's for cartoons DVR. It also depends on the demand, Right? Something like the Olympics or Superbowl needs a very strong service chain, I want to say, compared to a recorded movie on Netflix. Right?

Customers, it depends on the use case. If it's livestream, if it's let's say, you want to see the President of US, Trump speaking, then they better increase their reliability because if the president is, for example, giving a weekly speech or daily speech, they won't treat it as, they don't want to have as much unreliability as they would see in recorded stream. This is what QoS comes in, quality of service. Again, an element in the SP cloud that's not so common in the enterprise cloud.

Niki Acosta: Which increases the complexity and I guess the ...

Vikram Hosakote: Yeah.

Niki Acosta: I guess this is why there's a lot of use cases that can't just take out of the box off where from somebody. Like they're going to need something custom-built that works with everything they've done in the past, that works with how they do things in the future. Like they're trying to take legacy things then they can't have interruption. Right?

Vikram Hosakote: Yeah. That's a very good point. That's not something OpenStack can do out of the box and that's why OpenDaylight is taking up. Because OpenDaylight provides out of the box working service chains.

Niki Acosta: Got it.

Vikram Hosakote: It's for an SP and NFV customer.

Niki Acosta: Do you see a time when those are going to converge? I mean, you said you demoed it, but you think we're going to see a lot more of that in the future?

Vikram Hosakote: Yeah. I won't be surprised if a lot of customers want integration with OpenStack OpenDaylight. It's just that where ... It's a matter of time, right? An SP customer, like AT&T or Netflix, they want a bunch of virtual machines in a cloud like OpenStack. Today what they're doing is they deploy OpenStack and then they manually configure, like manually tie the two together. What we showed, it's possible in an automated way, yeah, it's certainly possible. I won't be surprised if the customer wants SP and a regular cloud, customer wanting the ... Sorry, an SP customer wanting an enterprise cloud.

Niki Acosta: Fascinating. I've heard a lot of estimates on just the number of devices and sensors and things that are going to be online, something like, was it a million devices per hour?

Vikram Hosakote: Yeah. The last time I read was a couple millions. Yeah. They're doing all sorts with them, matching them. I think not all of those don't use public IPs on the internet, but still a lot of devices.

Niki Acosta: It's a lot of data.

Vikram Hosakote: A lot of data. Yeah. It's a great point.

Niki Acosta: Muchos packets.

Vikram Hosakote: The smartphone, Twitter generation is fueling the SP world.

Niki Acosta: Those dang millennials.

Vikram Hosakote: Yeah.

Niki Acosta: They want everything right now.

Vikram Hosakote: Yeah. Because when we open Twitter, for example, we see a bunch of handles on the left called trending. That is conspicuous because they have to be so willingly receiving, doing analytics at a microsecond or millisecond level to figure out trends. Like yesterday, what was it, Rock talking to Alexa, was it? Or some sort of home automation device. Rock, the wrestler. That trended in two hours. When the lead singer of Linkin Park died, that trended in half an hour. How can they figure out what's trending when somebody dies in some country in some city somewhere without some sort of chaining going on? Right? They constantly blast or analyze Twitter data. They send this tricker blast to like an analyzing chain and then they come up with these all types of trends on the left vector.

Niki Acosta: Fascinating. Internet magic and the things we take for granted, all of course powered by the network. Whew.

Vikram Hosakote: Yes.

Niki Acosta: Fascinating. Well, we are about out of time. Before we go, is there anything else we should know about you? How do we find you? How do we get a hold of you?

Vikram Hosakote: Sure. I am available by email, of course. I work at Cisco, so you can contact me Cisco email. I am available in IRC. I mostly hang out in the Kolla OpenStack, Kolla docker communities and OpenStack Neutron channels. You can find my IRC handle, my email ID, my Cisco ID, my IRC ID, H-O-S-A-K-O-T.

Niki Acosta: Wait. Say it one more time.

Vikram Hosakote: H-O-S-A-K-O-T.

Niki Acosta: Hosakote.

Vikram Hosakote: Yeah. Right. No E. I think the email ID is ...

Niki Acosta: Oh, no. I may have messed that up.

Vikram Hosakote: I can send you my IRC. I'm active on Twitter. My Twitter handle is slashboot because I have two manage ... One of my very initial projects was to repair a boot director, slashboot director. I kind of did that and my Twitter handle is slashboot.

Niki Acosta: Slashboot.

Vikram Hosakote: Yes, slashboot. I'm not really active on Twitter. I know because once I become active, I will spend more time on it, and I don't have time.

Niki Acosta: Well, I hope that we can hear more from you because you have a way of taking very complicated things and making them sound not so complicated and we certainly appreciate that.

Vikram Hosakote: Yeah. Sometimes when I give talks or something I just Tweet about my talk. But other than that, I just IRC and email, and of course cellphone. Yeah. And Slack, if you're in Cisco.

Niki Acosta:	Awesome. Well, thank you so much for joining us. We had a number of guests. I'm not even sure who's next. I'm sure, I should know this, but we've got guests lined up pretty much to the end of the year .
Vikram Hosakote:	Oh, wow.
Niki Acosta:	Thank you to our listeners for joining us, everybody. Say goodbye.
Val Benincosa:	Thank you. Bye.
Vikram Hosakote:	Bye. Thanks a lot, Niki. Thanks a lot, Val. Goodbye. Have a nice day.
Niki Acosta:	Thanks, Vikram.
Vikram Hosakote:	Bye.
Val Benincosa:	See you, Niki. Bye.

For More Information

Find more Cisco Cloud Unfiltered podcasts.

Learn more about Cisco Cloud solutions.

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV Amsterdam,
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **www.cisco.com/go/offices.**

Printed in USA 08/17