EPISODE 694

[INTRODUCTION]

[0:00:00.3] JM: Virtual machines are operating system instances that run alongside each other on the same physical host. The virtual machines running on a physical host are managed by a hypervisor running on that physical host. A cluster of two physical servers could have four virtual machines running across those two physical instances, and those four virtual machines can communicate with each other over a virtual switch.

A network switch allows packets of bytes to be routed between machines. With a physical network switch, a dedicated physical device sits in the computer network to do this routing. A virtual network switch provides this packet routing without needing a dedicated physical hardware device for the routing. Open vSwitch is a distributed multi-layer virtual switch. Open vSwitch provides network switching for hardware virtualization environments.

Ben Pfaff is a core contributor to Open vSwitch and he joins the show to talk about operating system virtualization. Ben was also an early employee at Nicira, a company that made significant developments in software-defined networking before being acquired by VMware in 2012.

Before we get started, I want to mention that Software Engineering Daily is looking for sponsors. If you're interested in reaching over 50,000 developers, you can go to softwareengineeringdaily.com/sponsor to find out more. You can send us a message there. We'd love to hear from you. If you are an engineer that's working at a company that markets to developers, or if you're hiring developers, you can tell your marketing department or your recruiting department to go to softwareengineeringdaily.com/sponsor. That is one way to help us out, but also your listenership is quite good enough.

With that, let's get on with the episode.

[SPONSOR MESSAGE]

[0:02:01.1] JM: OpenShift is a Kubernetes platform from Red Hat. OpenShift takes the Kubernetes container orchestration system and adds features that let you build software more quickly. OpenShift includes service discovery, CICD, built-in monitoring and health management and scalability.

With OpenShift, you can avoid being locked into any of the particular large cloud providers. You can move your workloads easily between public and private cloud infrastructure, as well as your own on-prem hardware. OpenShift from Red Hat gives you Kubernetes without the complication. Security, log management, container networking, configuration management, you can focus on your application instead of complex Kubernetes issues.

OpenShift is open source technology built to enable everyone to launch their big ideas. Whether you're an engineer at a large enterprise, or a developer getting your startup off the ground, you can check out OpenShift from Red Hat by going to softwareengineeringdaily.com/redhat. That's softwareengineeringdaily.com/redhat.

I remember the earliest shows I did about Kubernetes and trying to understand its potential and what it was for. I remember people saying that this is a platform for building platforms. Kubernetes is was not meant to be used from raw Kubernetes to have a platform as a service. It was meant as a lower level infrastructure piece to build platform as a service on top of, which is why OpenShift came into manifestation. You can check it out by going to softwareengineeringdaily.com/redhat and find out about OpenShift.

[INTERVIEW]

[0:04:09.0] JM: Ben Pfaff is an Open vSwitch contributor and one of the designers of Open vSwitch. Ben, welcome to Software Engineer Daily.

[0:04:16.2] BP: Thanks for having me on.

[0:04:18.0] JM: I want to start with the topic of hypervisors. Explain what a hypervisor is.

[0:04:24.2] BP: I think of a hypervisor as something that lets you run multiple computers on top of one physical computer. It's pretty common that you want to run lots of operating systems, lots of little computers, but you only have a certain amount of hardware, or maybe you're a big cloud and so you want to rent out more computers than you actually own and subdivide them, and that’s what a hypervisor allows you to do.

[0:04:48.8] JM: To continue with the terminology, can you explain what the term network switch means?

[0:04:54.9] BP: Oh well, a network switch is what you use to connect up computers on your network. It takes a lot of network cables, either physical ones or virtual ones and then when a packet comes in, it figures out where it should go out.

[0:05:09.5] JM: What is network virtualization?

[0:05:12.3] BP: Oh, so network virtualization is the equivalent of hypervisor for your network. Usually, you only have a certain number of physical network cables, you only have a certain number of network switches, but you often want to subdivide that network so that you might have several tenants, you might have several customers, or if you're a large business, you might have multiple departments that you want to keep separate. They shouldn't be able to talk to each other, or only in very controlled manners. With network virtualization, you can subdivide your network in a controlled way.

[0:05:49.4] JM: Are there any other advantages of network virtualization?

[0:05:53.0] BP: Well, the big problem that it solves is right, subdividing and then controlled sharing. The other thing that it often does is it allows your users to control the network. If you go back a few years, then the means we had for subdividing a network was VLANs. VLANs are a limited resource. There's only a small number of them.

You have to have your network administrators do all the maintenance, all the updates, all of the adding and removing. With network virtualization, it's no longer a limited resource in the same way. You can give the power to set these things up to your users. You don't have to go through network staff, you don't necessarily have to take days or weeks to update things. The users can have more power and things can move more quickly.

[0:06:40.2] JM: Could you give me a brief history of how hypervisor switching in the Linux world has evolved over time?

[0:06:49.0] BP: Oh, sure. When you go back not that long ago, to I'd say roughly 2007 or so, there were only a few choices when it came to connecting the networks of virtual machines in a Linux environment. There were a few others that were niche, but the main one was what was called the Linux bridge. The Linux bridge is a software module, a Linux kernel module that acts like a software version of a network switch.

You could connect your VMs to the Linux bridge, and then you would connect the bridge to a physical network, and that would put your virtual machines on the network, so that they looked just any other device on the network. That proved fine for a while, but once you throw VLANs into the picture, the Linux bridge starts getting a little awkward. Once you want more advanced things like VXLAN and so on, you generally want things that are a little more advanced, a little easier to configure, maybe something that you can remotely configure, rather than just through CLIs. That's where I think of Open vSwitch is coming in. Open vSwitch is all those things. It lets you control things remotely, it lets you do more sophisticated things.

[0:08:06.5] JM: Talk more about the motivation for Open vSwitch and how the evolution of that Linux hypervisor ecosystem led to Open vSwitch.

[0:08:19.2] BP: I think of Open vSwitch as really coming into the picture when you're not dealing with an individual hypervisor, but you're dealing with a whole collection of them. You might have 10, you might have 50. When you have just one or two or a handful, it's easy enough to administer the networks of those hypervisors individually.

Maybe you don't need very much. Maybe you just need to connect them to the physical network. Once you start getting things in quantity, then you want more control over them. You don't want to be administering those hypervisors and their networks individually. You want a centralized controller where you can do all the administration for all of your networks in a single place and you don't want to even really have to care where those virtual machines are among your hypervisors, especially since virtual machines have this tendency to move around. You can move them from one hypervisor to another.

Open vSwitch runs in each of those hypervisors and then it connects to some controller that at least in a logical sense it runs centrally. Often they're clustered across a few machines in a distributed system. From the administrator’s point of view, they're only dealing with one entity. They're not dealing with 50, or a 100, or however many hypervisors they have.

[0:09:38.6] JM: There's a term I read about that is a little bit more abstract than Open vSwitch itself, and maybe we could just use this as a term to help people better acquaint with what Open vSwitch is doing. Can you explain the term it distributed virtual switch?

[0:09:55.4] BP: Distributed virtual switch, sure. When we talk about a network switch at the most generic level, you usually think of something that's in a single place. You think of something that you buy from Linksys, or Cisco, or Juniper and then plug cables into it. When you come to virtual switches, software switches like Open vSwitch, again at the simplest level, you're talking about something that's in a single place on a single computer, for example.

The thing is though when you have hypervisors, you naturally want to ignore the fact that they're on separate physical machines. You want to be able to pretend that that switch that you've got that's connecting your virtual machines, you want to pretend that it's spanned across all these hypervisors. That's a distributed virtual switch. It's a virtual switch that's distributed against across multiple hypervisors.

[0:10:50.6] JM: Let's say I have two physical machines. I've got two physical servers in my datacenter and I want to run for Linux VMs on top of those two physical machines, and I'm going to use a distributed virtual switch to be able to have those VMs exist; those four VM exist on just a couple physical machines. Explain the role of the distributed virtual switch in that scenario.

[0:11:16.8] BP: In a scenario like that, the role of the switch is to cleanly separate the network of the virtual machines from the physical network. You probably have some Ethernet-based network that's connecting to physical machines and you might decide for example, that you want all, or a subset of those virtual machines to be able to talk to each other, but you want their networking to be independent of the physical one, because for example, they represent some tenant that isn't entitled to access to your physical network in the data center. Or maybe two of them represent one tenant and two other ones represent a different tenant, and you don't want them to be able to talk to each other, except perhaps through some gateway.

In that case, the role of the distributed virtual switch is to ensure that separation. One of the most common ways to do that these days is through some tunneling or encapsulation technology. Open vSwitch support several of these and I think the best-known one at the moment is VXLAN. With VXLAN or another tunnel, you take these packets going between the virtual machines and you encapsulate them on the sender and then you decapsulate them on receiver.

The result is that these virtual machines, even though they're on a shared physical network, they have the illusion that they're on a network that they're only sharing with the other virtual machines in the same tenant. They can't talk to those other virtual, or physical machines that are on the network. They just have no contact with them.

[0:12:53.0] JM: If I were to spin up a fifth Linux VM on those same two servers where I've already got four Linux VMs running, describe in more detail, what would happen from the VM layer through the virtual switch layer to the hypervisor layer to the physical server? Give me a description of that provisioning process from end to end.

[0:13:14.8] BP: Oh, sure. The first question is which of these virtual or physical networks do we want the VMS to be attached to? Let's say that they should be part of one of the existing tenants. We have for example tenant A that had two VMs, we had tenant B that had two VMs. Now we're adding a third VM to tenant A.

The process that goes on here is add a few different layers. There's going to be some cloud management system that you use to create the VM and you use to assign it to a tenant. Then what happens next depends on the cloud networking system that you're using. There's a few different choices for these. The one that I'm most familiar with is called OVN. That's one that we've been constructing as part of the Open vSwitch project, although were likely to be separating it from OVS at some point.

Our goal at this point is for the CMS to tell the cloud networking system what to do. It will tell it something like, this VM is going to be on this hypervisor and you need to connect it to tenant A.

At this point, OVN will tell Open vSwitch that it needs to attach this to a particular bridge and assign it a particular ID. That ID is going to be used to put the packets into the encapsulation technology. Usually, OVN uses one called STT, or GENEVE, which is slightly different for VXLAN.

Then suppose that this VM comes up and it wants to send a packet, then that packet is going to go through a virtual network driver. It's going to go into the virtual neck. It's going to come out into Open vSwitch. Open vSwitch is going to look at the packet, it's going to run it through its flow table that OVN is set up and it's going to figure out where should this packet go. Let's suppose it's going to another VM that's in the same tenant, it's also in tenant A, but it's on a different hypervisor. If that's the case, then Open vSwitch will come across the flow that says take this packet, encapsulate it, so that it goes to the other hypervisor and that its source is marked as this new VM and its destination is marked as the other VM in tenant A and it'll send it up.

It'll get received on the other hypervisor and then you get the opposite process. It gets decapsulated, it gets identified as being for the VM. Open vSwitch puts it into the virtual NIC driver and that delivers it to the destination VM.

[SPONSOR MESSAGE]

[0:16:05.6] JM: DigitalOcean is a reliable, easy-to-use cloud provider. I've used DigitalOcean for years, whenever I want to get an application off the ground quickly. I've always loved the focus on user experience, the great documentation and the simple user interface. More and more people are finding out about DigitalOcean and realizing that DigitalOcean is perfect for their application workloads.

This year, DigitalOcean is making that even easier with new node types. A $15 flexible droplet that can mix and match different configurations of CPU and RAM to get the perfect amount of resources for your application. There are also CPU-optimized droplets perfect for highly active frontend servers, or CICD workloads.

Running on the cloud can get expensive, which is why DigitalOcean makes it easy to choose the right size instance. The prices on standard instances have gone down too. You can check out all their new deals by going to do.co/sedaily. As a bonus to our listeners, you will get a $100 in credit to use over 60 days. That's a lot of money to experiment with.

You can make a $100 go pretty far on DigitalOcean. You can use the credit for hosting, or infrastructure and that includes load balancers, object storage, DigitalOcean spaces is a great new product that provides object storage, and of course computation. Get your free $100 credit at do.co/sedaily. Thanks to DigitalOcean for being a sponsor.

The co-founder of DigitalOcean Moisey Uretsky was one of the first people I interviewed and his interview was really inspirational for me, so I've always thought of DigitalOcean as a pretty inspirational company. Thank you, DigitalOcean.

[INTERVIEW CONTINUED]
[0:18:25.1] JM: At a more abstract, or business application level, what kinds of network automation did Open vSwitch enable? Help people understand why is this actually useful?

[0:18:39.5] BP: Sure. I think of Open vSwitch as being closely related to the SDN movement in general. If you look at what was around –

[0:18:51.6] JM: SDN being software-defined networking?

[0:18:53.7] BP: That's right. Software-defined networking. If you look at what was around in 2007 and the challenges that people typically had at the time in their large networks, well a lot of it was simply that they had a lot of network devices. They had a lot of switches and routers and firewalls and other things and they had to be configured individually. This involved a lot of SSH, or, telnet or serial connections to these CLIs. Then you had to program these CLIs individually.

SDN came along and said, “This is silly. We need to centralize this management.” The initial attempt at this was what's called the open flow. With open flow, each of these switches, each of these network elements could connect to a single controller and the administrator could just change what the controller was set up to do and that would propagate it all out to these network elements. Since there's typically in a data center that involves lots of virtualization hypervisors, there are more virtual ports, there are more virtual switches than there are physical switches.

If you're going to do software-defined networking, you have to include your hypervisors in it, and Open vSwitch is the leading implementation of open flow, and so it enables that in a virtual environment.

[0:20:12.6] JM: You mentioned the year 2007. I'm guessing that's relevant, because that was around the rise of cloud computing, right?

[0:20:21.1] BP: From my point of view, that's relevant because that's when Nicira was founded. We were a startup based out of Palo Alto that was founded by a number of folks out of Stanford and Berkeley. Our goal was to figure out what to do with some research that had been done at Stanford and Berkeley around flow-based networking.

After talking to people who had big networks and big problems, we came up with the ideas that turned into open flow and that was later christened as a generic term, software-defined networking.

[0:20:55.5] JM: I see. Okay, so I didn't know you'd been part of Nicira. That was started by Martine Casado, right?

[0:21:00.8] BP: Yeah, that's right. I was the first employee there.

[0:21:03.9] JM: Very cool. What is the relationship between a software-defined networking and like in a cloud environment, so if I've got a cloud environment where my infrastructure is moving really dynamically and there's all kinds of requests for stuff to be spun up and spun down, why is it so useful to have the network to be virtualized?

[0:21:28.0] BP: Well, so suppose that you really just had a physical networking that we didn't have any form of advanced virtual networking, then what you'd have to do is you'd have to assign each VM a physical IP address, possibly one that was actually routable on the internet. You probably have to have somebody assigning these things manually. Basically without any form of virtual networking, it becomes very difficult to create and destroy and move around these VMs at any velocity. That fast-paced change is one of the hallmarks of a cloud environment, especially as containers start to come into the picture.

[0:22:12.6] JM: Describe the process of setting up a set of servers to use Open vSwitch. I've got my tenants and I need to configure them somehow with the hypervisor to use Open vSwitch, so that they can have this distributed networking system across them. What does that look like?

[0:22:35.1] BP: Open vSwitch is integrated into most of the hypervisors out there now. It works Linux KVM and Zen-based hypervisors. It works with Hyper-v. Usually, the process of integrating is really just a matter of turning it on in your operating system distribution, or maybe in installing the package. There's a small amount of say hypervisor-specific integration, but usually that work has already been done for you, unless you're doing something very unusual. Then to get it working in your networking environment, usually there's a small amount of configuration, usually you just point it to wherever you've installed your controller.

I'd say that installing that controller and getting it set up is probably more work. There's a number of them out there, and so it really depends on what you're looking for. There's open daylight, there's Ryu and fossa, there's OVN and all of them have their own installation processes. Of course, they're the ones that that come built into the various cloud management systems, like OpenStack has one that's built in and so on.

[0:23:41.3] JM: When Open vSwitch is running across these different physical servers, is there a consensus mechanism that runs between these servers to ensure that both of the servers running Open vSwitch have the same view of the world?

[0:23:59.3] BP: Open vSwitch by itself is not a distributed system. Open vSwitch runs on a particular host and it talks to a controller, or a set of controllers that you've configured. Now, it's really the controller that's responsible for making sure that there's consistency. OVS provides a number of low-level mechanisms the controller can leverage to help out with that, but the consistency is really the responsibility of the controller and the management system.

[0:24:29.0] JM: You said that controller, that is a per tenant abstraction?

[0:24:34.9] BP: Usually, it wouldn't be. Usually, the controller would have some understanding of tenancy. For example, if you're using an OpenStack environment, then OpenStack has a concept of tenants or projects, I think that they call them. Once you put that into OpenStack, it's going into a single controller that's part of OpenStack. Usually the tenants wouldn't provide their own controllers.

[0:24:58.5] JM: How does a system like OpenStack interact with Open vSwitch? Talk about that in more detail.

[0:25:05.7] BP: Sure. OpenStack has a number of agents that get installed on each of the hypervisors. In the case of when OpenStack is working with Open vSwitch, it has an agent for I think DHCP, it has an agent for different things that need to happen at VM startup. I believe it has some firewall agent.

OpenStack pushes information on the setup into these agents and then the agents push these things into Open vSwitch. If you're using OVN, there's a slightly different approach to it, but in any case, the configuration starts from OpenStack at the top and then gets pushed down to all the hypervisors according to what's actually running on them.

[0:25:54.3] JM: What's the value of Open vSwitch to a system like OpenStack?

[0:26:00.7] BP: If you look at the options that something like OpenStack has for networking, it has a few. One is that it can use IP tables and Linux bridge. In other words, it can use the mechanisms that are built into Linux. That is something that's very tempting for a lot of people, because they don't have to install anything extra. They don't have to make sure that there are any kernel modules, etc., that are going to work and be installed properly.

The disadvantages tend to be there too. Sometimes, the performance is not there. You also have to invent your own ways to control things remotely where that's built into Open vSwitch. Open vSwitch also has some nice properties in terms of how controllers work with it, how application developers work with it. It's designed, so that you can work with your application at the level of the application without having to worry very much about performance.

We've made OVS really good at taking the semantics at the level of the application and implementing them, so that they’re fast. That makes it a lot easier for some application developers. They don't have to worry about performance quite as much. There are other options too, like working via DPDK, working with VPP. These tend to be ones for people who are very concerned about performance. They tend to require a lot of customization. It can be hard to install and get these things working. They tend to use a lot of CPU, even when you're not actually having a busy network. Open vSwitch is designed to only use as much CPU as actually necessary for the network load at any given time.

[0:27:44.1] JM: Is it the case that Open vSwitch would also add similar value to other orchestrators, like Kubernetes, or Cloud Foundry?

[0:27:55.8] BP: Sure. If you look at Kubernetes, you can use it in a default mode where it uses things like IP tables. We also have a driver that's based on Open vSwitch and OVN. We think that that is pretty compelling. It has some performance benefits. We believe it has some configuration and management benefits. It's difficult to get the word out there, but we're working on it.

[0:28:22.1] JM: Well, tell me more about those benefits, like the configuration benefits and the management benefits.

[0:28:27.0] BP: Open vSwitch and OVN have a full array of network virtualization system primitives. It has switches and routers and load balancers and all of these things that you can't get in the stock Kubernetes networking. It has what I think of as a proven management system behind it.

I probably shouldn't try to go into too much of it, because it's not actually my area of focus. I should really defer on that to some of our other folks who spend a lot of time with Kubernetes. If I try to talk about it too much, I'm going to say some things that are wrong and silly probably.

[0:29:06.6] JM: All right, well I appreciate the adherence to correctitude. I don't know if correctitude is a word.

[0:29:14.0] BP: We'll go with it.

[0:29:15.3] JM: We'll go with it. I did read about something called logical tags. A distributed virtual which such as Open vSwitch can append logical tags to network packets. Explain what that means. Why is the logical tag important?

[0:29:35.0] BP: I haven't heard that exact term used, but I can guess what somebody was talking about and tell you what I think it's about.

[0:29:44.8] JM: Okay.

[0:29:46.1] BP: When you have a packet that's going between two environment, or going between hypervisors for example, sometimes you want to carry extra information that wouldn't normally be there on the wire. You might want to include things about for example, the context basically that the packet was received in, or sent in. You might want to for example, say something about the user, or the VM, or the process that sent the packet.

With GENEVE, with extensible and flexible encapsulation formats, there's a place to put that. You can add a field that says that information, and then you can include that as part of the processing, or part of the access control. Or perhaps, even if you don't use it in one of those ways, perhaps you can just pass it off to a monitoring system so that you can keep a better idea of what's going on in your network. Maybe you can learn more from an intrusion detection point of view, etc.

[SPONSOR MESSAGE]

[0:30:55.5] JM: Data holds an incredible amount of value, but extracting value from data is difficult, especially for non-technical, non-analyst users. As software builders, you have a unique opportunity to unlock the value of data to users through your product or service. Jaspersoft offers embeddable reports, dashboards and data visualizations that developers love. Give users intuitive access to data in the ideal place for them to take action within your application.

To check out Jaspersoft go to softwareengineeringdaily.com/jaspersoft and find out how easy it is to embed reporting and analytics into your application. Jaspersoft is great for admin dashboards, or for helping your customers make data-driven decisions within your product, because it's not just your company that wants analytics, it's also your customers that want analytics.

Jaspersoft is made by TIBCO, the software company with two decades of experience in analytics and event processing. In a recent episode of Software Engineering Daily, we discussed the past, present and future of TIBCO, as well as the development of Jaspersoft. In the meantime, check out Jaspersoft for yourself at softwareengineeringdaily.com/jaspersoft. Thanks to Jaspersoft for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:32:28.6] JM: Let's talk more about intrusion detection, or other elements of network security. What role would Open vSwitch play in network security?

[0:32:36.9] BP: Open vSwitch has some basic features for network security. It can implement a distributed firewall that does a stateful tracking of connections. It can definitely redirect packets to more advanced intrusion detection systems and so on. It can participate in NFV and service function chaining.

One of the things that we're thinking about, what we're pondering is should Open vSwitch be able to directly do things like switch on for example, the URL in an HTTP connection? That's something that's not there yet and we're trying to figure out whether that's important to our users and whether Open vSwitch is an appropriate place to put that thing. It might turn out that Open vSwitch isn't the right place, but we can do a better job of integrating with tools and systems that do integrate that, or that do have those sorts of capabilities.

[0:33:36.8] JM: It might be more appropriate to do that at a higher level?

[0:33:39.6] BP: Sure. Open vSwitch mostly does things on a packet-by-packet basis. These URLs and so on are you typically have to look at several packets in sequence and essentially buffer them up until you've seen the whole thing. There might also be a need these days to try to look inside SSL, or TLS connections using one or more methods, and Open vSwitch might not be the appropriate place to do those kinds of things. We haven't figured that out yet.

[0:34:06.5] JM: There are other areas of network monitoring that Open vSwitch is a useful layer for doing?

[0:34:13.7] BP: Well, Open vSwitch does have built-in support for a couple of network monitoring protocols. We support the older NetFlow v5 standard. We support the newer IPFIX standard and we support sFlow, which is a packet, a sampling protocol. You can configure any Open vSwitch instance to send packets to a collector one of those ways. We also support some protocols for network monitoring by you can replicate packets to a VLAN, you can replicate packets to an ER spanned tunnel.

[0:34:49.3] JM: Let's talk a little bit more abstractly. What are some of the design and implementation challenges of the Open vSwitch project?

[0:34:59.0] BP: I think that there are really two big categories of challenges. I'm speaking as a developer at the code level here. One is what are the reasonable ways to allow people to specify what they want and how much generality should we aim for? The question of should we support URL matching is the question there. If you don't support enough of this generality, then you won't be useful. If you try to cover everything, then your performance might suffer. That's always a tension there.

That actually leads into what I on a day-to-day basis, or I don't know, release-by-release basis is I think the number one challenge, and that is performance. Most of the things that Open vSwitch does are not hard. If you didn't care about performance, it would be easy to implement them. We want Open vSwitch to be fast, and fast enough to use in real systems and to have people be happy with performance. That's the big challenge.

We've written research papers about this. We published our different solutions that we've produced and we've certainly read and applied the lessons from a lot of other research papers. Sometimes, we get challenged by people who are building new systems that they've come up with new innovative ways to make things fast. I like to think of those as opportunities to find out how they did it and then try to use those same techniques our self, and the Open vSwitch support for DPDK is probably the most obvious item in that category.

When DPDK came out, it was a challenge. I felt like the developers there were showing off these benchmarks that made us look really bad, but we worked with them and we made use of their technology, we made use of their expertise. Now some of the DPDK developers are real OVS contributors working to make it faster and better all the time.

[0:36:58.7] JM: What else is the Open vSwitch community focused on right now?

[0:37:02.3] BP: Let's see here. There's a couple of things that are a little more practical. One is that we have what is almost two projects here; we have Open vSwitch, which is the virtual switch and we have OVN, which is a controller for network virtualization. They've always since the inception of OVN that they've been a single project mingled together. We're talking over how do we separate them, because the developers and the development techniques and so on are actually quite different, and it would make more sense to run them as separate projects.

Another thing that we're talking about, or at least I've been talking about a lot is the switch language called P4. A P4 is in some ways like open flow, but in other ways it's much more flexible. I'd like to get support for P4 into Open vSwitch, so that Open vSwitch can offer that flexibility to our users, to our developers, to academics who are trying to do interesting experiments and so on.

[0:38:03.0] JM: What are the other changes in networking infrastructure that you're seeing that are relevant to you as a core infrastructure developer?

[0:38:12.4] BP: Well, one of the interesting things going on in infrastructure is the rise of SmartNICs and the rise of NICs that can do a significant amount of offload. Open vSwitch works in software. One of the questions for performance is how well can we take advantage of hardware that's smart enough to do some of that work for us? We have done our best to work with NIC developers, and this is a whole wide range of capabilities and architectures in these. There's still a lot of question in my mind as to which of these architectures is going to win out? Maybe all of them will have their place in the industry, but I think there's some question about that.

In any case, we're not trying to pick a winner. We'd like to work with everyone and to make obvious good for all the hardware out there and all its capabilities. Of course, that's really difficult and we mostly have to rely on the contributions of the hardware developers themselves.

[0:39:13.1] JM: You've been in the networking space, I guess for more than 12 years. What has surprised you the most and were there any predictions you made maybe back in 2006 that turned out to be very far off? Did things deviate significantly from what you expected?

[0:39:33.6] BP: Oh, yeah. Yeah. One of the things that really struck us, especially in the early years was how hard it is to get some hardware developers and the industry as a whole to move in directions that seemed interesting. I remember we got a visit from a biz dev guy at one of the big networking chip makers, and we wanted to talk about how could we get some samples to experiment with, how could we figure out how to basically build smarter hardware switches.

The way I remember that meeting was the biz dev guy basically said, “Well, how many tens of thousands of these chips are you going to buy?” We said we're a startup. We're not just going to order a truckload of these things. He basically said, “Bye,” and they weren't interested at all in working with us. Obviously, things have changed a little as we grew bigger and we got acquired by VMware, but it's reflective of the networking industry attitude toward trying to do exciting new things, I think.

[0:40:36.2] JM: Any other changes you expect to see in data center infrastructure in the near future?

[0:40:40.6] BP: Oh, boy. Let's see. I've already mentioned SmartNICs, I've already mentioned P4. I think P4-based switches and other advanced hardware are going to be big. I think that that's really what comes to mind at the moment.

[0:40:55.1] JM: Okay. I found out at the beginning of this show you have a podcast around Open vSwitch. Mention again, what's the name of that podcast and how could people find it?

[0:41:03.8] BP: That's right. My podcast is called OVS Orbit. It's about everything relating to Open vSwitch and network virtualization and so on. It's a lot of academic talks and interviews with developers and users. You can find it at ovsorbit.org and on all the usual podcast sites.
[0:41:23.8] JM: Okay Ben. Well, thank you for coming on Software Engineering Daily.

[0:41:26.5] BP: All right, you're welcome.

[END OF INTERVIEW]

[0:41:30.7] JM: For all of the advances in data science and machine learning over the last few years, most teams are still stuck trying to deploy their machine learning models manually. That is tedious. It's resource-intensive and it often ends in various forms of failure.

The Algorithmia AI layer deploys your models automatically in minutes, empowering data scientists and machine learning engineers to productionize their work with ease. Algorithmia's AI tooling optimizes hardware usage and GPU acceleration and works with all popular languages and frameworks.

Deploy ML models the smart way and head to algorithmia.com to get started and upload your pre-trained models. If you use the code SWE Daily, they will give you 50,000 credits free, which is pretty sweet. That's a code SWE Daily.

The expert engineers at Algorithmia are always available to help your team successfully deploy your models to production with the AI Layer tooling. You can also listen to a couple episodes I've done with the CEO of Algorithmia, if you want to hear more about their pretty sweet set of software platforms. Go to algorithmia.com and use the code SWE Daily to try it out, or of course, listen to those episodes.

[END]