For Cloud Professionals, part of the On Cloud Podcast

David Linthicum, Managing Director, Chief cloud Strategy Officer, Deloitte Consulting LLP

Title: Delivering on the Promise of Serverless Computing and NoOps
Duration: 00:24:28

David Linthicum:
Hey guys, welcome back to the podcast. We're going to talk about something that's near and dear to my heart, and that's serverless computing and how it kind of relates to different things that are kind of emerging in 2019, specifically how it's going to relate to NoOps, which is kind of the big buzzword that's being bantered around right now. Ken – introduce yourself.

Ken Corless:
Sure. Thanks, Dave. This is Ken Corless. I am the CTO for the cloud practice here at Deloitte, and I spend most of my time working with our clients, trying to figure out not the *what* of the future but the *how* of the future, right? I think a lot of the technology trends and stuff, our clients understand. It's change-management and culture change at scale that is really the challenge. And, as we talk about serverless and NoOps, we'll repeatedly hear those themes, that it's not just about the technology – that people, process and culture are a huge part of success.

David Linthicum:
And we've got another very, very, very smart colleague with us today, Gary Arora. Gary, introduce yourself.

Gary Arora:
Sure. Hi, everyone. This is Gary Arora. I am one of the cloud native leaders here (inaudible 00:01:20), which basically means I'm a developer advocate for all things cloud native, serverless and open source. And I help our clients with their digital modernization journeys, including business case development, strategy, tooling, ecosystem selection, architecture, all the way to implementation with actual working software. So I'm really excited to be here and talking about one of my favorite subjects with serverless.

David Linthicum:
So let's kind of get into this pretty quick. So in other words we're dealing with a world today where we have kind of new and interesting ways in which we're building and deploying software. Of course we have container-based systems and we have traditional-based systems. Now, we have serverless based systems—the ability to, in essence, abstract ourselves away from the underlying infrastructure and not having to worry about sizing servers, which I always thought was a bit primitive. You know, when we build cloud systems going forward. Connect this to NoOps for me, Ken. So why is serverless important to NoOps?

Ken Corless:
Well, because I say that serverless, right, as we know, it's not really serverless, right? It's just about how much time and attention do you have to spend on certain components of your solution? So clearly, when you have a server solution, there's still servers. Just like cloud still has data centers. You just spend less time worrying about the data centers. And the connection between NoOps and serverless is that in a NoOps world – and I'm not sure we'll ever get to NoOps. One of my colleagues likes to call it LessOps. But the reality is, how do we spend more time of our finite resources developing capability that directly impacts our business, rings the cash register, delights our customers, rather than rebooting and patching servers and all that kind of stuff?

So serverless is a technology pattern that allows us to get closer to that NoOps panacea, where I could spend more and more of my time building capabilities that create business value directly, rather than doing the mundane operations of the world to keep the gears moving.

David Linthicum:
So Gary, do you agree with that?

Gary Arora:
Yeah. You know, that's one thing that is often kind of neglected, or takes a back seat in terms of just operations or security. For any new technology paradigm that comes in, it's all about the cutting edge, cool things it can do and then later, six months – you know, you realize, "Oh, the security and the operations of doing things the old way don't apply here." And that's why it's really important to kind of talk about these things, because serverless fundamentally changed how developers can approach building applications, but then it also changed how the operation piece kind of comes together, right? And whether it's NoOps, or whether it's LessOps, the idea really is that in a serverless environment, you've got to be – the choice is to be more intelligent about how your resources, hardware, software and people are kind of being utilized. It's about cutting the resource waste, associated cost and all of those things that serverless is doing for application development – kind of putting that also into the operations, the management, the deployment and those activities as well.

David Linthicum:
So Ken, we've always heard about the promises of automation around the ability to kind of, in essence, abstract ourselves away from very complex ops. So if we're moving to NoOps, how does automation relate to serverless computing, and what do we expect in 2019 as far as innovations there?

Ken Corless:
Yeah, I think if you look at what the hyper-scale cloud providers are offering in serverless – and by serverless we don't mean

just function as a service – things like Google functions, or AWS Lambda, right? We're talking about Kubernetes managed services, you know, various other containers – database as a service. And as Gary was saying, what the cloud providers are good at is two things. One, they're obviously good at automation and the engineering around that. And two, what they do is when they offer a capability, they tend to not offer it if it involves scaling human beings to create that for their customers. Because, again, they're in the business of hyper-scale. So their rigid attention to automation is really what unlocks the power of serverless, because you don't have to call somebody, you don't have to open up a ticket, right?

These are all the things that the developers, you know, don't like, especially in many of our large enterprise customers of the hyper-scale providers. They're still operating with yesterday's model where there's infrastructure teams who touch the infrastructure and the application teams who develop the code, right? And not having to have that level of interaction – one of the examples I always give in serverless is a great example. When you have a server, it has to be patched, right? There's no OS that doesn't have occasional security loopholes and other enhancements that have to be patched. The historic way of doing it is I'll have an application team who's working on something and the patching team knocks on their door and says, "Your stuff has to be patched. It's late." And they say, "It's not a good time, right, because we're having a real busy week. Can you talk to me next week?" And all of a sudden, we get further and further behind on patching, and we grow this animosity between two groups in IT—guys who want to get the patch done—who are measured on that, and developers who just want to code. In the serverless world, because there's no second group of humans, the patching is happening essentially in the background without disrupting anybody's schedule. There's nobody to request something of. And that model is really appealing to developers and the leaders of those developers who want to capture the incredible high velocity that they see young technology companies have, that in many cases our enterprise-class companies are yearning to get to.

David Linthicum:
So Gary, there's always an upside to any new technology. Serverless has a lot of upsides. Ken just went through them very well. In other words, the ability to kind of get out of having to maintain patches and fixes, and the ability to kind of automate and self-healing kinds of things, and the ability to kind of remove us from having to deal with the mundane tasks of operating these damn things. So Gary, what would be the downside, if a client asked you?

Gary Arora:
You know, that's great – I just published a paper on this a few weeks ago called Cons of Serverless, and I got a lot of feedback on that. Having been in serverless developing and architecting solutions, the general notion and the general sentiment that you see around the material, the collateral, that exists, is that everything is great and serverless is the next – you know, it's the next thing, better than sliced bread. But often, the downsides and the cons are neglected, and you only kind of realize that when you get to production or when you see some production-level loads coming up. And suddenly you see those benefits kind of slipping away.

But it's not so much as downsides of serverless as much as that you just need to be very careful in terms of selecting the stack, because there are so many options. And there's so much configuration that you need to like make sure you put it right, to actually gain the benefits. One of the biggest downsides, I would say is – and this is what I had on my paper as well, for which I got a lot of flak – was that it's actually expensive. If not set right, serverless can be more expensive than even some server-based applications, because all of this is now pay-per-use and all of these things that charge you based on certain allocation time. And so, if you have your application, which is kind of waiting for something to happen, but it's still kind of running, you're still paying for that, right? So these are like your API calls and things like that, which is kind of getting into details.

But there are many of these things, that you've got to now change the way you develop and change the way you architect to actually gain the benefit of serverless. Otherwise you just kind of take whatever was working in a VM or a container and you've just kind of transferred that to serverless. So you're not really kind of taking the full advantage. So the expense, the pricing is one of the key ones. The other ones, obviously you have your testing which changes dramatically. You have security, you have a little bit of your ops: you're deploying, your management, and things like that. And also, to a certain extent, you do have to kind of get into, not so much vendor lock-in, but you do get into a little bit of that tool lock-in. So whether you're using a Lambda, or a Google Cloud function, there are certain restrictions between them which kind of reduce your portability, if that is something that you are seeking.

David Linthicum:

Yeah, and everybody who's going to leverage serverless on a particular platform, they're going to be buying things down to the native APIs, correct?

Gary Arora:
Right, and as you start adding more bells and whistles to your serverless stack on any cloud platform, you start reducing your portability. Because you will not always find equivalent matches on other cloud providers, right? So if you want it for AWS, and if you are using Kinesis, SNS, and DynamoDB, and Lambda as part of your application stack, you can't move it one to one to another platform. You have to do a little bit of work-arounds. So that's the other thing that kind of doesn't – you know, it's not on the top of the list when folks are thinking of moving to serverless. But that's something that you've just got to think about while you're architecting your solution.

David Linthicum:
Yeah, it's okay to be devil's advocate of whatever architecture you're going after, because there's always upsides and downsides. So Ken, getting back to you, let's kind of back up a bit. And so say I'm a tire manufacturer in the Midwest, and I'm looking to move a lot of my traditional systems that are in my data center into the cloud, and I'm looking to implement a serverless kind of NoOps architecture as I'm making the move. What are the major steps and planning to ensure that I'm most likely to succeed in doing that?

Ken Corless:
I think the hardest thing to do is recognize that moving to a serverless architecture, which is typically associated with a loosely-coupled, micro-services-based architecture pattern, is not something that you can lift and shift to, right? We might have legacy applications that are fairly monolithic. You might have COBOL running on your mainframe or i/AS400 sitting in your plants. So the reality you have to first understand is, you know, getting to this serverless architecture, where do I get the most benefit? And I usually direct our clients to look at the areas of your systems that are undergoing the most change or competitive pressure in the marketplace with your digital strategy, and getting to the agility that is promised by a serverless architecture usually has the most benefit where you have to respond rapidly to business change.

I think looking at it and saying, "Hey, I'm just going to move all of my stuff to serverless," right? The reality is most companies can't afford to do that. The ROI for some of your most static applications just isn't there to leverage the new patterns. But certainly for some of your critical apps, and certainly for anything that's greenfield, I would look at leveraging the serverless architecture wherever you can. I think an important point to note is when we look at big, complex applications, often we're seeing hybrid, right? There are serverless components and server-full components, or whatever the opposite of that is. That when you're looking at the complexity and scale of some of these systems, that the reality is, you know, we're going to live in this hybrid world – cloud and on-prem, serverless and not-serverless, and figuring out how to optimize your infrastructure and your processes to live in what is going to be that hybrid world for at least a number of years.

I won't get to try to predict the future because of too many failed attempts at doing that. But the reality is the near future is going to be this hybrid world and understanding where to spend your attention to most get the benefits of serverless is really the key upfront analysis.

David Linthicum:
So Gary, where does the data live in all this?

Gary Arora:
Just to kind of tee-up on Ken's point, I 100 percent agree. The hybrid world is where things are moving. And one of the best or the most – you know, the biggest advantage of serverless is that it's such a plug-and-play model that it works with pretty much anything. Especially in the cloud native stack. I am right now implementing an application that is hybrid. It is using cloud native components like (inaudible at 00:15:01) search and also your Lambdas and your Dynamos and your (inaudible at 00:15:06) and your Kinesis for the serverless. And together, when you kind of put them together, you get the best of both with the pay-per-use serverless and the predicable usage that we have from (inaudible at 00:15:18) search, right?

And to the data point, data is also, by the way, part of the serverless stack. Not as popular as the function as a service, which, you know, kind of takes the crown in terms of everything serverless. But data, especially in terms of your storage, whether it's

database as a service or whether it's block storage with things like your (inaudible at 00:15:46), in the serverless model, that's when you actually see the biggest kind of benefits coming in, in terms of your pricing points. Because, in most cases, data is just sitting there, and you're not really reading it or writing it, so you shouldn't really be paying for that kind of activity. So by being able to charge you at a pay-per-use model, where if you read these many things, or these many frequencies, this much frequency in a month, then this is how much you pay, you're really able to kind of granularize and come up with really interesting payment models.

David Linthicum:
So Ken, what is the future of this stuff, going forward? You just participated in Deloitte's Tech Trends Report 2019. You know, that's out there now and a lot of people are reading it. What's the "Inside Baseball" scoop on serverless? In other words, where should the bets be placed by the enterprises out there? Should they sit on the sidelines? Should they jump in now? If they jump in now, what technologies are going to be most viable for them to leverage in terms of patterns and not necessarily brands? You know, ultimately where are we going to be in five years?

Ken Corless:
So, I say we're going to continue to see the hyper-scale cloud providers offer more and more managed services, serverless things, right? Look at what's gone on in the whole Kubernetes space just in the last couple of years, right? The reality is Kubernetes is actually fairly complex and involved to run. All of the cloud providers saw that and said, "Hey, so I just got you out of the server game, but we kind of have to replace your server team now with this Kubernetes team, and I'm living with a lot of the same challenges that I lived with in my server world. Because, quite honestly, most of our big clients moved to virtual servers five or ten years ago. So the reality is, trying to take out the levels of human interchange. I have a personal postulate – not quite certainly at the Moore's Law level, but that the productivity and agility of an IT function is inversely correlated to the unique number of people it takes to get anything done.

So I think when I look at serverless, it's about two (inaudible at 00:18:12) teams and all that. How do I get small teams able to fully manage/run, going at the core principles of DevOps? I think we'll see technologies continue to offer more and more serverless things, because a lot of the data center tacks of the past, patching and antivirus and intrusion detection, are all things that absolutely need to be done now more than ever. But to the degree I can make them silicon interfaces, rather than carbon to carbon interfaces—meaning people—the better chance you'll have of getting to the promise of what the C-suite wants, which is for you to act like these young technology companies and move at tremendous speed.

David Linthicum:
So question for you, Gary. So say we move ten years in the future. Ultimately, are we going to be at a NoOps state? Is this going to be a completely dark system? Everything's virtualized and we're at a point where not a lot of humans are spending time and worrying how to operate the system? I'm not saying we're going to move everybody out of it, but you know, ultimately it's a skillset we don't need anymore, and we don't have to deal with operations – everything's automated, everything's abstracted, and we're at a state where we just have solved this problem. Are we going to get there?

Gary Arora:
I think that's where we have been moving even, you know, over the past ten years. Throughout your evolution stages with moving to physical machines to virtual machines to containers, the idea was, "Let's abstract away a little bit more," right? And then serverless right now just happens to be the most current rung in this abstraction evolution. But yes, we're going to continue moving there [NoOps]. And that's a good thing, because that means the human time, the human hours, that you would now spend will be more focused on business logic, will be more focused on the core thing that really makes up the application, instead of all of the uncool things with your infrastructure management, and scaling, and provisioning, and all of those things.

There was a survey by RightScale, in 2018, where they said that the serverless penetration increased by 75 percent among the enterprises that they had surveyed, right? But so did all of the other ones too. Like container as a service or database as a service or SQL – database as a service, NoSQL, right? So all of these tracks that are moving, they all kind of – they are based on the principle of more abstraction. And I think that's where we are headed. And that's also going to transform into the DevOps world. DevOps is also changing. We cannot use the legacy ways of having a gatekeeper as a DevOps, where developers need to toss it over and then kind of wait until the DevOps, they get to it, and they get to deploy this in the right environment. I think

the market force is demanding much higher innovation cycle, or much frequent innovation cycle. We are moving into multi deployments in a week. Sometimes even in a day.

And those things are only possible when you take the human element out, and when the whole stack is automated. So now you can actually push multiple deployments coming in the same day, from different code bases.

David Linthicum:
So last question for you, Ken. Where does security play in all this, and what should we be considering and planning in terms of how security is going to exist in a NoOps world?

Ken Corless:
So I think security needs to absolutely be included, right? And many of the same principles apply, right? That the best way to protect an enterprise is not by the inspection of a security team who comes again and knocks on your door and is viewed as disruptive or even annoying to the application teams. The more I can put into automations, the intellectual property, and digitize the rules of security and make it be the only way to go. I'll give you a very simple, simple example. There have been a notable number of data leaks from companies who were using cloud storage and simply had permissions set by default that anybody who knew the URL could read it. Right? Most of those providers have switched their rules such that, hey, you actually have to go and explicitly make a decision to make something publicly accessible.

It's a very, very simple example, but that's a simple couple lines of code somewhere in the cloud provider's solution that dramatically improved the security posture of all of the people that are using it. Our old-world approach might have been to create a specialized team who provisioned cloud storage, and they were the only ones who were allowed to provision it, because they were the only ones we could trust to not make it publicly available, right? Which would slow down your development teams and all that.

So I think this notion of DevOps and automation, security and the aspects of security have to not be treated again as a separate thing, right? Many organizations have separate information security groups, but really pulled into the fold, and security is just another attribute of a solution, no different than performance, availability, or any of our traditional, non-functional requirements.

David Linthicum:
Great. Great discussion, guys. So Gary, where can we find you on the web?

Gary Arora:
You can find me on – depending what you want to do with it. So I have – I write a couple of open source pieces. I have authored two open source libraries. So if you want to know more about those, I am on GitHub under G-H-D-N-A. But if you want to follow me on Twitter, I also post about the serverless and cloud native kind of topics. I'm also on Twitter with @AroraGary and you can find me on LinkedIn with my name.

David Linthicum:
Great, how about you, Ken?

Ken Corless:
Twitter, @KFCWork. And I use LinkedIn also as well to publish some thoughts in various pieces, and occasionally if any of you follow the Wall Street Journal's CIO Journal, I occasionally have some pieces out there that are trying to address the concerns at the IT leadership level of our clients.

David Linthicum:
Well great. Great discussion, guys. Looking forward to continuing this at some point in the future. I guess we'll get together and powwow about what's proceeding in the space and what's worked and what's not. I'm sure opinions are going to be tweaked and changed, as they always are. But I'm looking forward to the next generation of this technology. Anyway, thank you very much.

Ken Corless:
Great, thanks Dave. Thanks for having us.

Gary Arora:
Thanks.

Operator:
Thank you for listening to On-Cloud for Cloud Professionals with David Linthicum. Connect with David on Twitter and LinkedIn and visit the Deloitte On cloud blog at www.deloitte.com/us/deloitte-on-cloud-blog. Be sure to rate and review the show on your favorite podcast app.

Visit the on cloud library

www.deloitte.com/us/cloud-podcast