EPISODE 1274

[INTRODUCTION]

[00:00:00] CQ: Hello, I'm Corey Quinn, Chief Cloud Economist at The Duckbill Group. I also host two podcasts myself; The AWS Morning Brief and Screaming in the Cloud, as well as write the perennially snarky Last Week in AWS newsletter, and I'm taking over hosting duties for Software Engineering Daily this week to take you on a tour of the cloud. Today, we're going to talk a little bit about Azure. And when looking for someone to talk authoritatively about Azure, you would be hard pressed to do better than Troy hunt.

[00:00:34] JM: A few announcements before we get started. One, if you like Clubhouse, subscribe to the Club for Software Daily on Clubhouse. It's just Software Daily. And we'll be doing some interesting Clubhouse sessions within the next few weeks. And two, if you're looking for a job, we are hiring a variety of roles. We're looking for a social media manager. We're looking for a graphic designer. And we're looking for writers. If you are interested in contributing content to Software Engineering Daily, or even if you're a podcaster, and you're curious about how to get involved, we are looking for people with interesting backgrounds who can contribute to Software Engineering Daily. Again, mostly we're looking for social media help and design help. But if you're a writer or a podcaster, we'd also love to hear from you. You can send me an email with your resume, jeff@softwareengineeringdaily.com. That's jeff@softwareengineeringdaily.com.

[INTERVIEW]

[00:01:37] CQ: Troy, thank you for joining me.

[00:01:39] TH: Hey, thank you very much for having me.

[00:01:42] CQ: You do an awful lot of things. I think the thing that everyone is most familiar with is the Have I Been Pwned data breach repository, which sends us some of the email that we'd like to receive the least. But you're also a Microsoft MVP, and a Microsoft Regional Director, which in true cloud naming fashion, sounds like you work there. Except you don't.

[00:02:04] TH: I don’t work there. I don't direct anything. I don't have a region. Any more questions, it is a little bit of a confusing title.

[00:02:12] CQ: How did you become the person that you are, for lack of a better term? You're well known in the data breach reporting space. You're sort of at the forefront of a lot of infosec discussions these days. But it's easy to fall into the trap of assuming that people have just sprung fully formed from the forehead of some gods somewhere. What's your origin story?

[00:02:31] TH: Look, to be honest, it's a bit of trial and error and a little bit of falling into the pit of success. I think it would be reasonable to say in some ways. Look, my background as a software developer, I started building software for the web back in the early days in ’95. I still got my HTML for Dummies book on the shelf behind me, which is where it all started for me. And then I went through a very sort of traditional, I guess a traditional role for those of us in the tech era, went through .com times, which of course was a bit non-traditional in many different ways. But then I went to work for Pfizer Pharmaceuticals. Now, of course, these days, everyone knows who Pfizer is, because they're making a vaccine. Back then I used to say, “Hey, if you don't know who Pfizer is. You know what Viagra is? Yeah, that's us.”

So I spent 14 years at Pfizer, originally, as a software developer. In fact, I was building classic ASP, and then .net. And then I went into an architecture role, which was basically the thing you do if you want your career to progress, rather than the thing you do, because you actually want to do it. And in Pfizer world, what that meant for me is I looked after the Asia Pacific region, and anything that we built software-wise in that region was something that I got involved in. And this was now around the late noughties, I want to say, 2009-ish kind of era. And I really wanted to drive us more towards modern compute paradigms. I know that’s a bit of a broad term, but I really wanted to get us not just out of the on-prem-centric world, but I wanted to get us out of the like virtual machine, run your own VPS environment. We'd have instances of IS up on someone else's hosting. I wouldn't even call it cloud yet. And we'd have an instance of a database. And these would be machines that required TLC. And I wanted to move us to cloud, but I wanted to move us to cloud paradigms in cloud.

And I really remember that I think the point where it really clicked for me is we had a new solution being built, and the vendor was like, “Alright, we want a virtual machine for it. And this is just a website, right?” I'm like, “Well, why?” “Well, because that's what we always do.” I would really like to push us down a PaaS route. I was very, very enthusiastic about PaaS.

And Have I Been Pwned was born pretty much in equal parts to be. Data breach service is predictable, but also to demonstrate as Azure. And I just wanted to go and build something on Azure, because I was now an architect. I wasn't getting to build stuff. And frankly I kind of miss that. So I was like, “I'm going to build something on Azure. And I'm going to demonstrate how this platform works. And I will use cloud paradigms such as an app service instance in Azure.” I don't have my virtual machine. I just literally have a hosting service. So we use the database service. I will use something really, really way out there. I will use a table stories to put my data in rather than putting everything in a great big relational database.

And it's funny, like we're approaching eight years on now. And some of this is actually starting to feel, “Oh, and in fact, a bunch of what I have replaced with newer cloud paradigms. But that was the thing that drove me at the time. And through that those efforts, I managed to then drive a bunch of Pfizer onto Azure and onto our platform as a service as well.

[00:05:45] CQ: It's hard to find customers who are doing interesting things with Azure that aren't exactly the opposite of what you just talked about. The idea of, “Yeah, I'm going to build this virtual machine, just an extension of my data center.” It looks an awful lot like what I would build in a physical facility, except now I'm not going to trip over a power cable and knock everything down. I can pay people to do that at scale for me. It seems that the people I know who are really, other than doing toy problems on top of Azure, are mostly the large blue chip companies that tend to move slowly, because they have this risk aversion toward sizable sums of revenue, “Yeah, we're going to throw it all away and build it from scratch again, because why not?” doesn't seem to be something that large banks and insurance companies tend to buy us for.

You’re sort of a shining example of using closed cloud native principles in a context where, within Azure, it seems that there aren't a whole lot of notable case studies of similar folks doing similar things. Do you feel like you're doing this alone? Or am I just completely looking in the wrong place?

[00:06:45] TH: It's probably not the right question to ask me only insofar as because I travel in circles, which are very Microsoft-centric. In a year, I do see other organizations doing various things with Azure. But I also think if we sort of scroll back and we look at what drove me in that direction and what might cause other organizations to take a more traditional approach, there are a bunch of things working in my favor.

So one of the things working in my favor was I didn't have any money. Because I didn't have any money, like I had to be really, really cost conscious. So I made decisions that were best supported by new cloud paradigms that were available very, very cheaply, but required you to write for that paradigm. So let's say table storage. Table storage was the single best decision I made in the architecture Have I Been Pwned. And it's a very, very cheap, so far, in my experience, infinitely scalable resource that is also very responsive. It does a lot of other things terribly. If I wanted to join across tables, it's an absolute nightmare to do that. It's just for all intents and purposes impractical.

Now the challenge I think that many traditional organizations have is they've got these embedded processes and embedded expertise and they're to reuse that across modern cloud paradigms. And this goes back to the example before where this vendor just wanted to have a virtual machine. And they wanted to put things on a SQL server running on a virtual machine. And we would always sort of joke about. And it's funny, I haven't thought of it for quite a while now. But it was always that old thing of, "Oh, if you want to store data, you will need a SQL server database." It's like that's a very, very powerful piece of equipment. Are you sure? Maybe it should even just be like literally a text file on a file systems somewhere depending on what your needs are. It's like it's a bit of the old if all you have is a hammer, everything else looks like a nail.

So I think the sort of incumbent organizations with incumbent software and incumbent skill sets and thinking had a lot of trouble implementing these new paradigms. And I had the benefit of just having a clean slate and frankly, the benefit of having curiosity as well.

[00:08:57] CQ: I hear you. As far as looking at something like a SQL server where it's great, I need something that's incredibly powerful. I'm going to pay a license on a per core basis. And

that's great. And then you have on the other side of the world where my preferred database is, of course, Amazon's Route 53, in which I misused DNS txt records and use it as a key value store in exactly the wrong way. Now, sure, we all have our ridiculous peccadilloes around those things. But it's an upgrade from my old approach of storing a SQLite file in S3, checking it out by every function that was running it, making whatever changes it needed, and then putting it back where it came from. Professional advice, never try this, because it ends exactly as poorly as you would expect as soon as you're introduced to concurrency.

You've been around for a while, but one of the things that really, I think, put you on the map was the early days of the Ashley Madison breach, where I believe that was the big watershed event for you, or am I misremembering the breaches? Lord knows there have been so many, where suddenly you were subject to massive spiky traffic.

[00:09:57] TH: Look, it's interesting, because when I look back, there's no sort of single one incident for me which defines it. Look, if anything, it was probably January 2019, with the Collection #1 data, which we might come back to a bit later. But I think the thing with Ashley Madison was, yes, there were large volumes of traffic. But frankly, the all the memories of Ashley Madison were more about the subject matter, and the sensitivity, and the outcomes of that breach, the outcomes, in some cases being suicides. That's what really sticks in my mind.

I think traffic-wise, it's funny, actually, when it happened, this is a great sort of cloud advocacy piece here. I was snowboarding with my kids. And I'm like down on the snow like – Yeah, we do actually have snow in Australia for listeners. I was down at the snow in August 2015, enjoying the time with the kids, and this whole thing broke. And I just remember like sitting on the chair, going up the chairlift, taking my gloves off, pulling out my iPhone, and using RDP into a server to then spin up new instances of cloud. And I was like, “How freaking cool is this?” I'm on a chairlift in the snow, like literally spinning up the cloud in order to respond to, what at the time, I think probably was unprecedented traffic. But it's like, “That's cool. I can just spin up more instances. And it's all good.”

[00:11:13] CQ: Yeah, it's easy. Just throw more hardware at the problem. It goes away.

[00:11:17] TH: Yeah, exactly. And this is mostly good, mostly. Here's where I'm starting to hesitate. Because what I found with Ashley Madison, and in fact, what's driven some of the architectural design decisions subsequently, is as an enamored as I was with the whole PaaS model, PaaS is still logical units of compute of a fixed size and a fixed cost. And you're just adding these logical units. And in my mind I was like you're just adding blocks. It's like I'm adding more blocks to the puzzle. And every block is the same size. Just a question of how many blocks? How much of the block am I using? Do I need more blocks. And in fact, the experiences that I was having was particularly with things like auto scale, like I was only getting more blocks when the capacity the other blocks ran out. And that meant when the capacity ran out.

[00:11:58] CQ: Yeah. Auto scaling is great. It gives you exact capacity you need 20 minutes after you need it.

[00:12:03] TH: There's a bang on, bang on. And this is why so much of the really critical stuff is subsequently rolled over to functions and serverless. It's like, well, just take my money. Like take as much as you need. Give me as much as I need. And have me no longer think about the whole concept of these blocks.

[00:12:19] CQ: I love the concept behind serverless, where functionally what you're doing is paying per request more or less. And it's a very tiny fraction of a penny per each request. But it also leads to interesting economics insofar as it's not just, generally speaking, less expensive than traditional approaches. But it also ties your model of economics back to whatever your usage pattern looks like. So instead of being able to say, "Oh, it's going to cost me a fixed," whatever your data center **[inaudible 00:12:43]** bill would have been once upon a time. Instead, it's, "Oh, every thousand users on the site is going to cost me X-dollars or X-cents," or whatever it is that it works out to. And at least for me, from my perspective, that's always turned into something as a lot more predictable as a function of growth or as a function of traffic.

Now, some more traditional finance folks get very concerned because they want to be able to predict to the dollars and cents. And, well, it's great that you saved us some money. But can you just go ahead and build a data center, because we know what that's going to cost? It'll be fine. Feels like that is not exactly a growth industry at the time of cloud these days.

[00:13:16] TH: No. It almost feels like a perverse incentive, doesn't it? I know that accountants are necessary in life just like lawyers. I these people have to be there. But they sure make life difficult sometimes. And that the challenge even back in the PaaS days where we'd say, “Look, we're going to get this instance of PaaS. It's going to be on a on a logical app service. We can actually have many, many instances of PaaS on this logical app service. And when we get to capacity, we can just add more instances.” And they're like, “Well, what do you put in the spreadsheet?” “Well, we don't know, because we're not there yet. If we're really successful, there'll be more.” And they would struggle and struggle and struggle with this concept.

In fact, I remember it got so difficult at Pfizer that one of the guys on my team was literally paying the Azure bill off his own credit card, which was around about $75 a month. So it wasn't crazy. But it was so difficult to actually get costs reimbursed. We were saving something like 20 grand a year, and he had to wear the $75 a month on his own card.

[00:14:11] CQ: I think professional advice if you’re listening to this, don't do that. Don't finance your company. Money should really flow one direction. It's for them to you. But I digress. Please continue.

[00:14:20] TH: You know what? I'm having flashbacks in recent times. Because I have an API key that's $3.50 a month, which is just – Like I basically just pegged it at what does it cost to get a latte at my local coffee shop. And the idea is to be just enough to keep the bad guys out. And the amount of trouble I hear from people just trying to get their $3.50 a month reimburses have gone like – This is why I'm independent. Part of the reason I'm independent.

[00:14:43] CQ: I’m right there with you. It's one of those, “Okay, you have a hard cap of $100,” or whatever it is before you need to get two levels of approval in various companies. But then you look around and you can call a meeting with how many people multiplied by their average hourly rate means you spent how many tens of thousands of dollars today? It's cutting off one's nose to spite one’s face.

[00:15:00] TH: I get these emails from people, and I'm just so tempted to go like, “You idiots. What are you doing?” And then it's like, “This person was me before. Like I have sympathy for you, mate. I'm really sorry, you're in this position.” But too bad, figure it out.

I think sort of going back to that serverless piece as well. One of the other interesting incentives it creates is when you have such a direct correlation between the performance of the code that you write and the cost that you incur. In my case, it's still just me sitting here running for it and paying for it myself as well. That makes things really interesting because you really, really start to critically analyze things not just from a performance for user experience perspective, but from a performance for cost perspective. And I just love that direct correlation between how efficient you can be and what cost you in terms of paying for the service.

[00:15:51] CQ: You were involved in Microsoft's ecosystem for a very long time by modern cloud standards. Not to age you prematurely. And that means that you saw it back in the days when it was still called Microsoft Cloud, if memory serves. It was the very early stumbling steps that the entire industry was more or less making toward this idea of cloud computing. What was that like? How did you see that evolve? And what did they get right? What did they miss?

[00:16:16] TH: So a fun story. I only recently had to retire an article of clothing that was still branded Windows Azure. Now I'm going to say what this article of clothing. And just everyone keep in mind, this means something different in Australia to what it does everywhere else. So I have, or I had, Windows Azure thongs. Now depending on where you are, you might call them flip flops, or other things that you –

[00:16:37] CQ: I was going to ask as my next point of disambiguation there.

[00:16:40] TH: You might call them flip flops or sandals or something. You put them on your feet. But yeah, I actually had Windows Azure thongs. And every time I'd see them, I think back to, “Remember when it was Windows Azure?”

My very early memories there, and we sort of come a little bit full circle here in terms of the enterprise and their adoption of cloud, is that the very early offerings of Azure we're not consistent with the existing offerings that we were used to from a web hosting perspective from

Microsoft. And what I mean by that is that we've had ISS for a long time, and you could create websites, and you could create your classic ASP, or later your ASP.node apps, and you could push them into the website, and they would run. And when we got those first offerings, and I'm scratching my head, I'm trying to remember exactly what it was called. But it was not like lift and shift sort of stuff. You couldn't just take your apps and run it there. You literally had to go, “Okay, I'm going to create a brand new app.” However, I guess you could reuse some HTML and some JavaScript. But basically, you're rewriting a bunch of server-side code, A. And then B, you're working with a paradigm, which is completely unfamiliar to absolutely everybody.

So that was a barrier for us at Pfizer. We were literally just investing more in shared hosting. Literally buy our own VPS and slice it up and put all our things on it. So that was keeping us from moving. And Microsoft really had to get to the point where they're like, “Okay, well, this thing that we've actually been making work for the last couple of decades, we're going to now offer this on the cloud.” And that became the Azure App Service. And finally, we actually had the ability to move over something that would work equally on our VPS environments as it was on our new cloud paradigm environments.

[00:18:17] CQ: It sounds almost like you were stumbling over a lock-in story, which was not around technical capability so much as it was the patterns of the people who were in place working on these environments, when the only environment you've ever been exposed to looks like a traditional data center running a three-tiered application. It's very challenging to move beyond that as individuals and far more so as a group. How did you get there?

[00:18:39] TH: Well, and I mean, this is, I think, sort of a fascinating side of cloud. And it's very easy for us – In fact, a fascinating sort of technology in general. It's very easy for us to get very, very focused on the tech itself. So what does this tech do? What are the upsides? What are the downsides? And in doing so, I feel that we often miss what are the human aspects of implementing this. And what I mean by that for us at Pfizer is no one understood it. We didn't have anyone in the organization that understood it. The vendors, which we would use didn't understand it. So as good as the technology might have been from a pure tech perspective, if we don't have people that can work with this and understand it, yeah, A, that's a big barrier. B, if we start pushing them down that direction, we were starting to outsource everything to vendors. That's another podcast there. We had problems with that. Do we want to be paying our vendors

to learn something new on our dime, and then also be paying them to actually build software on the new thing as well? And then for the folks internally, it's like what is the right balance between actually delivering working software and sending them off to learn something completely unfamiliar?

And look, I think even now, that's still a problem. But I suspect that particularly as we've moved away from more monolithic sort of structures to more augmented hybrid things made of many parts and many different skill sets required to do it, and maybe it's not as much of a problem now, but particularly going back really a decade and a half, that was a big challenge for us.

[00:20:08] CQ: It feels like the hardest part of shifting organizations, the technology is there, whether we want to admit that or not. But the challenge, of course, is throwing away this, I guess, “legacy”, 40 years of revenue generating things and effectively giving a lot of that over to a cloud provider. Given the experiences that you've had, and the companies you've talked to, and the types of environments you've been in, how do you, I guess, make the case or seen the case made to go ahead and actively trust the third-party vendor to run the thing, which is really what cloud has always been about? We've been lying and say, “Oh. No third-party vendor could take me down.” Well, never mind the fact that you have a single power outage, a power line coming into the building. But now it's more explicit, because when you're running it in someone else's cloud environment, it's very clear that your uptime is in many respects in their hands.

[00:20:59] TH: I think you sort of touched on something really interesting there, which is around the narrative. So what is the discussion we should be having about cloud? And one of my early recollections was I would get asked, “Is the cloud secure?” And this is a very, very Boolean question. Someone's looking for a yes or a no. And what I’d always sort of say to people is it's differently secure. There are aspects of cloud, which I believe are going to be much more secure than our traditional approach. One of those aspects is that if we go and put things in Microsoft's App Service, it’s going to be run by very, very dedicated professionals. And there are a lot of them, and they ever see each other's work. And they have all sorts of levels of experience and processes that we just simply don't have even as one of the world's largest companies, because, hey, we make Viagra. Like we don't make cloud. This is the difference in our MOs.

And when it comes to things like tripping over the power cable, I mean, that's another great example. When we talk about availability, there are differences in availability. You are at the mercy, if you like, of the cloud provider in terms of what they do on their end and what availability they managed to achieve. But if you're not, and you are running the whole thing on-prem, tripping over the power cable? Are there power failures? And then are they redundant power supply failures? And is there outage on internet connect – Like there's all these other things that can go wrong?

And the discussion we got to have is to look at these two things next to each other and ask the question of how are they different? And then collectively, is this something which tips the risk or the ROI into one side or the other? As opposed to this sort of massively simplified Boolean question of, “Should we go to cloud or not go to cloud? Or is it secure or not secure?”

[00:22:41] CQ: Well, the security point is really a great place to take this conversation, given that you are, at least today, most broadly notable for the Have I Been Pwned service. It's integrated into one password and a bunch of other services as well. It winds up effectively telling you whatever email address you have given it. It has been shown up somewhere on a list of breached credentials, which is really been a great story, for one, stop reusing passwords, folks. And two, making sure that you're aware at least when these things happen. Do you find that the security story where we're now going to effectively trust one of these third parties, in your case, Azure, outweighs the security benefits of more or less running it yourself?

[00:23:21] TH: I think the security story, again, is just a different story, as opposed to there being a clear tip of the balance one way or the other. But here's where I think cloud has created a lot of risk. And it's not that Microsoft or Amazon or anyone else has weak implementations. And let's say for argument's sake, people are continually breaking through the security of S3 storage and pulling all of your data out. That's not the problem. The problem is, is that all of these organizations have made it so cheap and so easy to put huge amounts of information in the cloud that anyone can do it. And it has never been cheaper and easier to screw it up.

This is why we had a few years ago, there like so many MongoDB instances, which were publicly accessible without a password. We've had a lot of S3 buckets. We've had a lot of other modern cloud paradigms that have had vulnerabilities, not because of – I'm going to rephrase it.

It’s not vulnerabilities. They've had data exposures, not because of vulnerabilities in the product or in the hosting provider, but because any idiot can now get themselves cloud. I say that very, very bluntly.

[00:24:31] CQ: It’s a single place to look for – When you're looking for exploits, it’s a single giant target that has an awful lot of surface area. And you can start iterating through and enumerating these things. And it's a different story as opposed to dealing with how everyone's bespoke on-premises NAS or SAN has been configured. This is, well, you know where the starting point is, the ending point is and you can iterate through incredibly rapidly.

[00:24:53] TH: Well, it does make it very easy. And, again, it's sort of like all these things about cloud, which we love so much from a software development perspective can make it a nightmare security-wise. It does have high availability. It does have high bandwidth. And a lot of the time it is publicly-facing as well. So, well, that makes things easy, doesn't it? Like it makes it easy to stand stuff up and build wonderful products. It makes it easy to expose it with insufficient security configurations as well.

[00:25:20] CQ: There is, of course, the argument to be made as well, that one problem you don't have to worry about, if you go into the whole shared responsibility model of how clouds talk about security. Things like physical security are not a concern. There have been notable breaches where data has been stolen from physical data centers by someone driving a truck through a wall. I know it happened at least once in Chicago, for example. They grab a rack into the back of the truck and peel out of there.

I get the sense that when you're dealing with a tier-one public cloud provider, that's not really a threat vector anymore. I trust that Amazon and Microsoft and Google are not going to let hard drives out of the facility once they have entered that facility without rendering them down into pieces. I don't have to worry about people buying my old database drive on eBay and then doing a recovery on it.

[00:26:06] TH: I don't think any of those things are realistic concerns with any of the major cloud providers. I do think they're realistic concerns, especially with organizations running their own intro. But I do wonder how often that actually makes an appearance in the conversation people

have around if it's time to go to the cloud or not. And even going to the cloud, we say that in a very general sense as though it's some sort of bully in position. And, of course, it's much more nuanced.

[00:26:33] CQ: Oh, yes, hybrid cloud is always a story of people who started moving to the cloud and got stuck halfway along the way, or it's taking a lot longer than they thought. And at some point, they had to call it something else.

[00:26:42] TH: Well, I think the premise of hybrid in so far is we don't have to do everything in one go. We can do bits and pieces. And we can do just the things that make sense is great. I think things like the recent Hafnium situation with Exchange. Keeping in mind, I don't work for Microsoft. I think that was a great advertisement for Office 365. I'm sure that didn't quite work it out that way. But there must have been someone who sat there and went, "Oh, well, there's probably going to be an upside to this. We might roll some more people over to cloud." But that's a great example of where we can take one part of an organization and we can move something like exchange to the cloud, and perhaps keep a whole bunch of the other stuff local.

[00:27:21] CQ: Yeah, the idea of what workload makes sense to migrate versus which are going to take a little bit longer. You did mention again that you don't work for Microsoft. So I'm going to lean into that particular aspect. One of the things that I find that is a little odd to me is everyone talks about, "Oh, AWS is number one. And Azure is number two," when we look at the Gartner surveys and the rest. That's great. It's fun to hear that in the abstract. But then I look into my customer accounts, I look at what I see out there, and very few companies that I've spoken to have ever had a serious conversation of, "Well, time to go to the cloud. Do we go to Azure? Or do we go to AWS?" I do see some of those debates happening between GCP and AWS much more frequently. But Azure has been one of those things that seems like it's a little bit different than that.

[00:28:07] TH: Look, it's a good question. And I think maybe the observation here is that the different cloud offerings in some cases are quite well aligned. And other cases tend to be much more bespoke. I guess one thing that does separate Microsoft from the other players is that they have such an incumbent audience that's running their operating systems internally within the organization, obviously, especially on desktops, but as well as their server environments.

[00:28:30] CQ: Oh, they have a terrific ecosystem.

[00:28:32] TH: Yeah. Well, that's certainly got many decades of having done this, and that that's an advantage that they have over some of the other providers. So there's, I guess, somewhat of a more native fit there. But then we look at things like modern app services. And there's much less incumbent use that's tied to a stack there. Overcoming, of course, that sort of natural tendency to check everything on a SQL Server database that I think inevitably this is where things do differentiate a bit. And just by virtue of the fact that when you go into most organizations, you will log on to a Windows PC and authenticate to an active directory controller somewhere gives Microsoft a foothold for parts of the service, which I don't think Amazon and GCP even necessarily want to compete on.

[00:29:18] CQ: Different areas and different strengths. I mean, I use this line from time to time, and people think that I'm trying to be intentionally insulting, and I assure you, I'm not. But one of the big benefits that Microsoft has in a time of cloud is that they have 40 years of experience apologizing to businesses for software failures. And you need that in cloud, because it's computers. It's what they do. They break. There's no way around that. And they're very good at articulating the value of, “Well, everything just fell on the floor, but it's okay,” to executive stakeholders who are not themselves very often deep in the technical weeds. And I don't think that that is a strength that is appreciated enough in the common discourse.

[00:29:59] TH: Yes. I mean, I think the way you put it in terms of having decades of experience photos and people, it was very interesting. But, again, when you do look at availability, and you do look at outages, and frankly, across all the major providers, it is rare enough that when it happens, obviously, it's high impact. But when it happens, it's suddenly a major, major story, “Look. How massive is it?” Let's even take someone like Netflix. Like how massive of a story is that when Netflix is out particularly in these modern times when everyone's at home? So it would be interesting to look at what sort of availability levels do we get now compared to, let's say, pre-cloud days?

[00:30:39] CQ: I think on some level, there's also a bit of a transformation. If you go to Google to search for something, and the site doesn't load, or your Netflix stream suddenly stops

streaming. I don't know about you, but my default assumption there is, “Oh, my WiFi went out.” And I assume it's going to be a problem on my end on something either in the WiFi or what's going on with my ISP, as opposed to the other side of things where it's, “Oh, no, no. It's fine. Everything you're doing is great. But Netflix is down, or Google is down.” These things effectively never happen anymore. That took a strange, I guess, pivot in some ways. But for better or worse, it feels like we talk about utility computing. A lot of the services have gained that aspect of utilities where whenever I flip a light switch or turn on a faucet, I don't wonder what's going to happen next. It's expected if it ever doesn't happen. Well, wow! Something massive just shifted.

[00:31:32] TH: Well, firstly, we're in Australia. So it's always our Internet. That's always the problem down here.

[00:31:36] CQ: Yes. Insert joke of Telstra here.

[00:31:39] TH: Oh gee! Let's not go down that rabbit hole. But I think that whole sort of utility view of cloud as well. I mean, this was – If we think back – yeah, let's think back maybe a decade and a half. This was the value proposition. This is going to be a commodity that we turn on and we use as much as we want. And just like a commodity, such as electricity. I think we're sort of now at this point, as you mentioned, where the expectation of always having availability is now so high that suddenly we get a bit of a shock if it's not there. It's not the natural assumption to blame the cloud if something isn't available. You start wondering if the PC is still plugged in.

[00:32:19] CQ: Yeah. On some level, the failure mode is not, in some respects, running out of capacity. It's “Huh. Wow. Okay. It's scaled up as I told it to, and it served all the traffic. And that's awesome.” Especially in the serverless world where you have things like functions that just fire off and scale infinitely, except there is a constraint, and that constraint is budget in many cases. And one thing I will say that Azure your gets very right, that AWS gets hilariously wrong, is there free tiers actually legitimately free. There are no stories that I have seen on Azure just saying, “Oh, you're on a free tier as someone who's just learning how this stuff works. And, oh, here's a surprise $7,000 bill.” But that happens almost monthly on an AWS basis that makes it into

public, never mind the private conversations. If I'm sitting in my dorm room trying to learn how this stuff works for the first time, I don't want to accidentally have just bought beliefs.

[00:33:09] TH: But this is what concerns the accountants. So, I mean, we had that discussion a bit earlier on about accountants want to sort of see fixed cost and predictability and all the rest of it. And they see these stories. And this is what I think in many cases draw us and to say, “Look, we need predictability.” And the real tragedy out of that is that you lose the opportunity to realize so much of the benefits. I don't want to see people getting surprised bills. And I think that there are a whole bunch of particularly UX-centric paradigms that we can use to avoid those surprises. But I really want to see people only using what they need. I mean that's fantastic for everything from the bottom line to the environment.

[00:33:47] CQ: Oh, it absolutely is. I think this is one of those areas that transcends whatever cloud provider we're talking about at any given moment. The myth is that you pay only for what you use. But in practice, you're paying for what you forget to turn off. Very often, one of our cloud economists exploration voyages into a customer account, it's, “Okay, you have this extra petabyte of this giant cluster sitting here in the developer environment. What happened?” Well, it turned out the developer who spun that up or copy that data off no longer works at the company. And cool. Yeah, there's no garbage collection in cloud accounts for better or worse. These things will retire after you do. Left to their own devices.

[00:34:26] TH: Yeah, I'm sure there have been a few surprise bills there. Which then, of course, brings us back to the discussion about this is one of the reasons organizations with their procurement departments make this stuff so hard to try and get more control over things like this?

[00:34:41] CQ: Oh, yes. And there's a security element to this too. If you aren't aware that these things exist in your environment, but you're certainly paying for them, who's updating them? Who's validating that their security posture and data accesses are still within bounds of what they should be? And on some level, with at least the reason that I focus on, the economic side, instead of the security side, is that no one calls me at three in the morning freaking out about a cloud bill most of the time. That is strictly a business hours problem, whereas security is the exact opposite of that. And, frankly, I enjoy getting a good night's sleep. Though, now that I have

a seven month old, I used to enjoy having a good night's sleep. Now I just lie there and wait for the next screaming from one side to the other.

[00:35:22] TH: I hear you. I hear you. I've been there. Understand.

[00:35:25] CQ: Now, it's a hard problem to solve for. I don't know what the right answers are. I do think that Azure was something that was, to be direct, a little bit of a joke five or six years ago. I don't think it is nearly as laughable now as it was then, especially when we start looking at the broader Microsoft ecosystem. I think that done right, the future of cloud is really Microsoft's to lose. With GitHub being the de facto answer for everyone who keeps their code and VSCode being what people are learning to write within for the first time, it really seems that they're just a deploy to Azure button click away from effectively owning the future. But they need to shore up some of their platform fundamentals first. And if they wait too long, the opportunity passes. The bias of timing is super critical here. But I am very bullish on the future of Microsoft and cloud. And for someone who grew up hating Microsoft from the open source side of the world, they've turned new corner, definitely. It feels ridiculous to hold that against them now 20 years later.

[00:36:25] TH: Did you use to spell Microsoft with the dollar sign?

[00:36:27] CQ: I was never quite that far gone just because it seemed like – That's a little too edgy in my parents' basement style. I prefer to be a little bit more eloquent, as far as crapping on large companies, especially these days, where crapping on large companies seems to be my stock and trade.

[00:36:42] TH: I think what's sort of interesting there with the GitHub observation as well is that the entire ecosystem around cloud has expanded so much. I mean, if I get back even to those early days of Have I Been Pwned, it was very much around, "Look, there is platform as a service. And you can put your things here." You build them all locally and you publish them up very often in a fairly manual process. But now there is so much other stuff surrounding it. And if I'm honest, there are all these terms that I see popping up as, "Oh, well. I was sleeping when that – How did I miss that thing?" There is so much periphery around the process of building and deploying and managing software over and above the software itself. I mean, I find that

both fascinating. If I'm honest, a little bit scary, because I start to realize I'm missing a bunch of stuff.

[00:37:31] CQ: Yeah, and that's the problem too, is watching the surface area expanded everything. Where we talk about, “Oh, only –” What is it? 20% of application workloads have moved into the cloud. Well, we kind of have a lot. That percentage has gotten relatively large, especially in light of how big that pie has gotten. And it feels like everything is exploding out geometrically. Who can say that they're an expert on all things cloud or in all things one provider?

[00:37:58] TH: Normally, very self-ingratiating people. I would say it. Hey, maybe that's an age or experience thing as well. I find myself being much more comfortable these days for saying, “I don't know. I don't know –”. To be honest, even the way I'm deploying at the moment at GitHub into Azure, I'm pretty sure I'm not doing it the right way. And I'm okay saying that. But this is sort of part of the learning experience too. And what I still find really good fun even now is people will work it out. I'm in my mid-40s. I still really enjoy being able to sit down and learn something completely new that I just haven't seen before. And there're so many opportunities to do that. So I could look at it as overwhelming. I could say, “Hey, what a cool time, right?”

[00:38:41] CQ: You say that you're convinced you're not doing the deployment right from GitHub to Azure. I don't think I've ever met anyone who has said that, “Oh, yeah, the way that we take our code and deploy it is spot on great.” Everyone says what you just said. Where I'm sure I'm not doing it right, but it kind of gets there in one piece the way I intended to. So knock on wood and hope it doesn't break. May I ask what you're doing?

[00:39:05] TH: Sure. So there're different services within Azure that I'm using. But if you pick, say, the App Service, which is what runs the haveibeenpwned.com website, when you first learn that front page, it's using the Kudu service that runs on the app service environment to grab check-ins in GitHub, pull the code, compile it, deploy it, and it deploys it to a staging environment. And then it automatically swaps slots between staging and production.

Now that is like literally doing the compilation on the app service environment. This is something that I could be using – I think I can use GitHub pipelines or Azure. I don’t even know what the

proper name is. It's where I am at the moment. I could be doing it in an external environment and then pushing in that compiled code. And then if we go on – I'm even trying to remember how I'm deploying functions. I think it's the same way. I'd have to go back and check my configuration, because I don't think I've ever deployed that for like a year or something.

I know that – And I'm like literally airing on my dirty laundry here. But I know I'm at a point at the moment where it's like I would have to think very carefully before doing a deployment of a lot of things, which I really don't want. Because I want to have this sort of laissez faire attitude of being able to, "Y'all, I just run deployments whenever. Everything should be stable and predictable and repeatable." But I think the reality of it is, particularly when you're a one man band in this case, as time goes by, you end up a little bit scared to actually touch the working things.

[00:40:34] CQ: Oh, my stars. Yes, the things that are working. And once you've haven't touched it for three to six months, forget it. Completely forget it. It may as well have been written by someone else. Like, before this recording, I spent an hour or two working on some old legacy code to extend it in some interesting ways. And oh my God! Previous me was the worst developer on the planet.

[00:40:53] TH: Well, I think, at least in my situation here with Have I Been Pwned, it's like what happens if it stops working for a while? It doesn't really matter. Like it will come back up. There's sort of bits of it I'm a little bit more worried about things like the pwned password service, where there's just like thousands and thousands of services out there doing password checks on like login or registration. If that went down, I'd be a little bit more upset about it. But, then again, so what am I going to do? Like refund the free money that they're paying in order to use the service? So, look, I just do my best. And it usually works.

[00:41:29] CQ: For better or worse, it seems to have worked out for you folks. For what it's worth, I've never tried to pull up Have I Been Pwned and found it down. And I think that that's something that also gets as people lose sight of it, where if a customer doesn't try to hit your site, or a user has been trying to access your site during a period where you've taken an outage. Where you really down? It's one of those questions for the ages.

[00:41:52] TH: Look, there's also – I keep coming back to like these middle ground positions, which sounds a little bit evasive at times. But it's not always up or down. There are some times where there are bits that just don't work as well. So one of the things that's worked exceptionally well for me is CloudFlare. So it's very hard to talk about cloud and not include the role that CloudFlare has played with Have I Been Pwned. So when I talk about something like pwned passwords, in fact, I was just pulling the stats here before. So this is the k-anonymity-based model that allows you to see if a password is appeared in a database without actually sending the password. So if I look at, say – I had what I think is probably the biggest day yet the other day. We had 36.85 million requests in a day to that service, which is really, really cool. I'm getting close to a billion a month at the moment actually.

Now, of the 36.85 million requests, CloudFlare served 36.81 million from cache. Now, might there have been an outage at the origin server at the time? Possibly. But we've managed to reduce the risk of anyone actually experiencing an outage by literally distributing around hundreds of edge nodes of CloudFlare around the world, which is great. And now in just the same way, the front page of Have I Been Pwned is really heavily cased at CloudFlare. So even if we have a situation where, let's say, the domain search feature at the moment still runs on the app service, and particularly for larger domains, it can become quite slow, it can cause the app service to scale, it can lead to timeouts and unavailability of that particular service. But the front page, it's sort of the same app service, gets cached at CloudFlare. And then when you literally type your email address in and that search box and click search, that goes to a functions API. So it doesn't hit the app service. So they can be bits of the service that are slow, or even out completely. And there are other bits of the service that are just performing beautifully.

[00:43:49] CQ: It's a distributed systems problem for better or worse, a small toy site for lack of a better term, because this architecturally, one person feels it. It feels similar in some aspects, at least spiritually to a toy site. Even in these relatively small scale environments, it turns into a distributed systems problem, where increasingly it's not a question of is the site down? But rather, how down is it at any given point in time? CloudFlare's various edge locations across the world, a couple 100 of those. it's a near certainty that at any given point, at least one of them might probably be having a problem of some sort. But how can you see it? How can you know? And this is, I think, where the whole observability world starts to look at these things and pop-up with a bunch of vendor-provided solutions.

[00:44:33] TH: Well, the question then, of course, is that that scenario that you've painted earlier on where something's not responding, and you immediately sort of blame your own internet connection, that then, of course, makes things very nuanced, where we now moved from the point where we had a very small number of closely physically related dependencies, and things that could go wrong. To now, we have got stuff all over the place. There are a lot of different places that stuff can start to get wrong. And we get higher availability and all sorts of other funky things that come with that as well. But when we get to the point of like trying to figure out where is the problem now, that does make for much more interesting troubleshooting than what I did in the past.

[00:45:15] CQ: I will say that for better or worse finding how – Find out where the issue is when you have five or six functions talking to one another and you wire these things together has always been, I guess, the big sticking point for me, tracing claims to be able to solve this. But at the same time, instrumenting applications for distributed tracing is often a bit of a heavy lift. It turns every outage a murder mystery though. So there is that

[00:45:38] TH: I had one just last week, someone who is using one of the services, which has an auth layer around it. Said, “Look, the querying an email address.” So there're some enterprisey services where you can query an email address by hash prefix using a k-anonymity model. And they said, “When we're querying this very occasionally with the correct API key, we're getting back 403.” And I'm like, “Oh gee! Where do I even begin with this? Does it happen all the time? Does it happen close to –” And I'm thinking, “Is at CloudFlare? Is it Azure? Is it their app services? Is it the function? Is it the underlying data?” And we kind of went round around over a period of weeks because it was extremely rare. Until we got to the point where they went, “Oh, we found the problem. It’s our code.” Troubleshooting this stuff is now becoming so much more complex that even when it’s at their end, it can actually be quite difficult to get to the bottom of it.

[00:46:36] CQ: It's one of those areas where, for better or worse, it feels like if you're holding still in technology, that events can easily outpace you. As someone who started off working in the hands-on keyboard sysadmin style role, what's your impression of how Azure has

progressed? Does that person still have a clear path forward? Or is it time for some serious upscaling?

[00:46:57] TH: I think the constant challenge with Azure and probably with the other cloud platforms as well, is that there is constantly so much new stuff. So I used to joke. I mean, sometimes I do a conference talk. Back in the days when we used to stand in front of people and talk, and I do a bunch of live demos, and I pull up the Azure dashboard. And I'd literally look up at the big screen. I'm like, “Ah, that wasn't there yesterday. I wonder what that is? That's something new.”

And I think the challenge that we've got here is that there's so much constant innovation with new paradigms that are wonderful and are doing cool stuff. But you end up with it's almost like this mental weight of, “Geez! Look at all the stuff here that I don't know, that I don't understand. What does that do? Could that do things better than what I'm doing at the moment? Am I missing out by not using this new thing?” I think that that's just an inescapable challenge. And frankly, for me, I just gotten to the point where I was, “Ah! It all works pretty well at the moment.” If I have to go in and learn something new, then I'll do that. But I just need to accept that I simply cannot stay on top of all of the new things.

[00:47:59] CQ: The idea of just-in-time-learning. On some level, it's important to be open to the idea of learning new things and not the easy grudge of, “Oh, I'm going to just crap all over anything that doesn't tie directly into my view of the future.”

[00:48:11] TH: And that brings us back to that original sort of enterprisey position of, “We would like the VPS, because this is what we understand. Please give me my server now.”

[00:48:20] CQ: And it's even harder to fight that tide when you have 5000 engineers working at a shop who would all have to be upscaled. But if they all know how to work with this in the legacy paradigm, it's hard to say no sometimes.

[00:48:31] TH: That's true.

[00:48:32] CQ: Thank you so much for taking the time to suffer my slings, arrows and strange Azure questions. If people want to hear more about what you're up to and how you're thinking about these things, where can they find you?

[00:48:41] TH: Troyhun.com, or Twitter @troyhunt. Pretty much everything starts from one of those places.

[00:48:46] CQ: Excellent. I will of course toss links to that in the show notes. This ends today's tour of the cloud. If you enjoyed this podcast, follow me on twitter @quinnypig. And, of course, head on over to lastweekinaws.com and subscribe to hear more from me on my podcasts, The AWS Morning Brief and Screaming in the Cloud, and of course the snarky Last Week in AWS newsletter.

[END]