EPISODE 1460

[INTRODUCTION]

[00:00:00] JM: WP Engine is a domain-specific cloud provider that hosts high-performance WordPress infrastructure. This website, Software Engineering Daily, runs on WP Engine. Scaling a domain-specific cloud provider for WordPress includes complexities at the level of the database, application load balancer, and other areas.

Brandon DuRutte is from WP Engine and joins the show to talk through his work.

[INTERVIEW]

[00:00:25] JM: Brandon, welcome to the show.

[00:00:27] BD: Thank you. Happy to be here.

[00:00:29] JM: You work on WP Engine. And this is what I would consider probably the best WordPress hosting system, or at least up there. And I think people might have the perception that WordPress is easy to run. It's been around forever. Simple application frontend with a database attached to it. But in fact, there are lots of complexities to making a scalable WordPress deployment system. Could you outline some of those complexities? What makes scaling WordPress hard?

[00:01:06] BD: Yeah. And I appreciate the compliment there about WP Engine. I agree that I think we're one of the best. And we've put a lot of engineering effort into building our platform to be very scalable and very performant for customers of all sizes.

When I think about sort of scaling applications in the cloud, or modern application architectures, I think about the 12 factor apps kind of approach that Heroku engineers put out years ago. And these are principles for application design around how you manage state, how you deal with deploying the software. And WordPress doesn't really follow those principles. It sort of predates those principles.

And so, for instance, there's in-app upgrades, and you can install the plugins and themes from directly within the running app. And the consequence of that is that state about what the software that's running needs to be shared by any replica that you have of the application. And so we, at WP Engine, as well as other platforms out there, no doubt, this is a core concern. It's sort of a tradeoff between the flexibility that you give to your end user in terms of do I want those features and the performance or the scalability of the site?

And we at WP Engine tend to skew that choice in terms of giving builders flexibility in terms of how they build their site, how they manage their site within WordPress. We give them the flexibility and take on the burden ourselves of ensuring that we can scale up and scale down, that we have a fast performant file system, that we have caching in front of it that ensures that most of the traffic doesn't actually require a PHP execution, which is the language that WordPress is built in. And of course, we also integrate with CDN partners to provide the static content as close to the end user as possible, as well as provide things like DDoS protection and other things that, if you're going to do it yourself, these are all kind of concerns that you would have to worry about just to get the kind of scale that we have at WP Engine.

[00:03:29] JM: What are the constraints of having an application that is purely built around PHP?

[00:03:37] BD: Yeah. PHP has come a long way, I would say. Constraints wise, the PHP runtime’s execution is single-threaded, as opposed to something like a Node.js, where it has sort of process switching embedded in it. So you get a request for a PHP execution when the like a database query is being executed. The PHP code just waits, and that process or thread can't go on to do other work while it waits for IO that's happening in the database. So you have sort of a more constrained memory usage, because you have to keep that process waiting for the IO. That's sort of the big difference between PHP runtimes and other modern web stacks.

[00:04:33] JM: Can you give me a sense of the deployment model for a given application? So like, my website, Software Engineering Daily, runs on WP Engine? What is it being deployed in?

[00:04:47] BD: Yes. So I didn't go look in our systems to see exactly where you're being deployed. But our platform has undergone an evolution. Likely, you're on a more or less a LAMP stack. So we have MySQL, Apache with mod PHP, a Varnish Cache in front of that to

provide page caching. And then we front it with nginx that allows you to have rules around redirects and rewrites, as well as efficiently serve your media content that's stored on the disk. That's the core of our stack. And then, of course, the CDN in front of that.

[00:05:27] JM: And so what are the places where you can leverage the economies of scale, the fact that you're running tons and tons of WordPress, rather than lots of ad hoc applications?

[00:05:41] BD: Yes. So I think there's some things definitely in our caching configuration that applies across a variety of WordPress environments. Definitely, most websites on our platform have very bursty traffic patterns. So very few websites are getting significant traffic all the time. And obviously, when you get up into the top 100, top 1000 sites, you're getting constant traffic.

But in the main traffic is very bursty. And so our platform is able to sort of share the resources across multiple different WordPress sites. And we have proprietary sort of technology to make sure that no site is monopolizing resources. And that when your site gets traffic, it's going to get served efficiently and quickly.

[00:06:43] JM: Part of the advantage of using WP Engine, and this is not like a sponsored advocacy, but you have like a lot of tooling around failure modes. So if my database gets corrupted, or if somebody logs on, hacks my site, detonates my site, you have a lot of backup infrastructure and a wide range of other cron jobs and stuff that are running. Can you tell me more about the platform that surrounds the WordPress installations?

[00:07:19] BD: Yeah. So that's absolutely right. We have daily backups that run nightly depending on which geo you're in. Because we want to make sure that you always have that backup point, at least one day, 24 hours. Of course, as a user, you can definitely create additional backup points if you're going to make changes to your site or something like that.

We also do – We manage updates for WordPress. So we have systems that monitor for WordPress releases. And when those releases come out, we validate that those releases are good. Historically, we had one incident where a version of WordPress came out that was incompatible with our platform. So we don't want to ever inflict that on our users. It's one of the reasons why we recommend disabling the auto update feature in WordPress.

Certainly, we validate that. We will do upgrades. We'll ensure that the upgraded site is working before we officially complete the upgrade. We also have a similar technology for doing plugin upgrades. We do some database maintenance, kind of cron jobs, on a daily basis. We do tracking of analytics and logging, and present that to the users and our user portal. Lots of stuff like that. So we have cron jobs that run and do hourly kind of jobs, and daily jobs, and then weekly jobs as well for various kind of maintenance purposes.

[00:08:54] JM: Are you built entirely on AWS?

[00:08:59] BD: No. In fact, we use both AWS and Google Cloud, as well as – So the flywheel platform, which is a part of WP Engine. Also, some WordPress is running at Linode and also at DigitalOcean. So we're a multi cloud operation.

[00:09:16] JM: Is that for resiliency? Or are there services that you get out of Google Cloud that you don't get out of AWS?

[00:09:25] BD: It's not specifically for resiliency. It's more a business decision not to tie ourselves to a specific cloud, and to architect our systems in a way that makes it portable between clouds.

[00:09:39] JM: Gotcha. What's the advantage of that cloud portability?

[00:09:43] BD: I think in the past, we have had incidents with not the major clouds, but with hosting partners in sort of previous generations. And so, in order to protect ourselves sort of in a disaster recovery kind of situation, not sort of real-time resiliency, but in that kind of a situation, we would have portability across the different clouds.

And from a business perspective, it's definitely an investment, a significant investment to build it that way. But it also provides us some – It provides us some leverage and some, I guess, flexibility for our customers. Some of our customers have requirements that their websites operate on certain clouds. We tend to not want to tie ourselves to a specific cloud where we don't have to. But if we have a customer where it's an absolute requirement, we can migrate them to a place that satisfies their business requirements.

[00:10:43] JM: Are there any places where you have to put in additional security guarantees on top of the simple authentication systems that WordPress has built in?

[00:10:58] BD: In our platform, we definitely have additional security features that prevent many of the common drive-by WordPress attacks. So when a user authenticates to our platform, we cookie them in a way that our systems sort of know that user's permissions and what they're allowed to do sort of outside of WordPress, right? So it offers that authenticated user permission to, for instance, upgrade plugins, whereas a drive-by users simply can't do that. It's not even possible to get that level of access to the file system.

Other security features we have, for customers who are very security sensitive, we have a product called Global Edge Security. This is a CDN integrated WAF that prevents all traffic from reaching our origin host without being inspected in and validated against a set of security rules that prevent malicious traffic. So lots of layers of security in our platform, for sure.

[00:12:04] JM: Tell me more about those – What are those drive-by WordPress? What are the common WordPress attacks that can occur?

[00:12:10] BD: Yeah. So you'll hear from time to time, or you'll read maybe, a WordPress plugin vulnerability discovered 600,000, websites vulnerable. And a lot of times these are some kind of SQL injection or cross-site scripting thing, or other kind of security vulnerability built into a plug in in the ecosystem. The exploits for those cause can be used to deface or hack websites.

And basically, they're exploitable without any – In some cases, anyway, without any authentication. So you can get authenticated access without actually logging into the website. And so our platform actually is resilient to a lot of those kinds of attacks just because of the way our security systems are set up.

[00:13:05] JM: How do you handle updates to WordPress? The open source project gets updated all the time? Do you force those updates on your user base?

[00:13:14] BD: We generally try to keep our platform up to date, keep sites on a current version of WordPress. If you fall too far behind, you lack security updates and other things going forward. We don't necessarily force upgrades. We do upgrade everyone on our platform. But we

allow you to opt out of those upgrades for a period of time. We support back versions, but not indefinitely.

[00:13:43] JM: Tell me about what happens when a website is getting like massive traffic and you have to scale up the site and do load balancing.

[00:13:56] BD: Yes. In general, when a customer gets a massive amount of traffic, depending on the way the site is built, first, we'll scale up sort of vertical scaling, adding just more compute resources to it. In most cases, that's sufficient. If you do have a website that has a significant amount of traffic or is expecting a significant amount of traffic, we do have a plan that does have replicated webheads, we call them, behind a Cloud Load Balancer with a database that's a managed database. So it's a little bit different than our main, the architecture that supports the majority of our sites. But it does allow it to auto scale up and down and is a sort of fully managed environment that will handle that scale.

[00:14:49] JM: There are people who use WordPress for different things. So some people just use it for basic website hosting. Other people use it for e-commerce. Are there different deployment schemas that you need to issue to handle the different use cases?

[00:15:08] BD: That's a great question. So in general, our platform is pretty good at handling a variety of use cases. Commerce is one where, in particular, because it has sort of direct economic ties, we do have an offering that has additional capabilities, in particular, around product search. That's a really big one for WooCommerce stores or WordPress commerce stores. And the search inside of WordPress isn't all that great for that use case. It's using database search technology by default, which is it’s full text indexed. But it doesn't do fuzzy matching and other things.

So when you have an ecommerce store, you really want a better search experience than that to allow your visitors to find the items that they really want to purchase. So that's what our ecommerce plan is targeted at.

Most of the other use cases – In fact, all of the other use cases, we run on the same WordPress stack without much difference. I guess the one exception to that is our new product line, Atlas, which is a headless environment. So this is for people who want to run their frontend in node.js,

perhaps a React app, or a View app, something like that, and then only use WordPress as content management system with an API in front of it. So we have a platform that allows that kind of deployment model. And people are using that for media sites, corporate sites, that kind of stuff, because it does allow for a little bit better throughput and, yes, scalability, depending on how you build your site.

[00:16:59] JM: When you talk about the product search modifications you need to make, what's your backing system for search?

[00:17:11] BD: Yes. So we have a partnership with ElasticPress, which is a WordPress search technology built on top of Elasticsearch. So ElasticPress, Elasticsearch, it can be a little confusing. But, yeah, it's at the WordPress layer on top of Elasticsearch.

[00:17:28] JM: Gotcha. And do you have to – If I'm running some ecommerce store with a massive product catalog, do you have to do anything complicated to modify the – Or do you have to expose any configuration stuff to the user through your own front ends? Or just the native ecommerce system together with the integration take care of anything? Do you have to build any specific frontends for it?

[00:17:56] BD: You don't have to. You certainly can build things like faceted search to search on sizes, or colors, or items. But much of that is plug and play in WooCommerce. Getting your catalog into the search, that's relatively straightforward. It also allows you to do things like waiting for items in your catalog. So if you're trying to feature certain items in your catalog, you can give them preference in the search results, things like that. But in general, it’s is pretty plug and play.

[00:18:31] JM: How has the deployment system changed over time? I think you guys are using Kubernetes now. Tell me about how that's evolved?

[00:18:42] BD: Yeah, that's right. We have begun a migration onto Kubernetes. So in the past, we were very VM centric with configuration managed VMs. The next sort of step of the evolution of that was actually running containers directly on those VMs. Sort of setting ourselves up for migrating all of the workloads into a Kubernetes environment where Kubernetes is doing the

orchestration and the scaling and that kind of stuff. So yes, we have taken our platform from a VM architecture into Kubernetes.

And one of the things that we did during that that's maybe worth calling out is we're moving away from Apache with mod PHP as the PHP runtime. And we're moving to using PHP FPM, which is the FastCGI process manager for PHP. That gives us a little better resource utilization. In particular, that was necessary when we moved to Kubernetes just because of trying to make better use of our resources so that we can have more memory and other things for scaling out.

One consequence of that, downstream consequence of that, is people who are using htaccess rules, which are available on our core platform. We're having to help them migrate to a new rules engine that we built to manage those kind of rules that happen that are sort of pre-processing before PHP takes over and runs your code. So that's the migration path that we're on.

[00:20:18] JM: Can you tell me more about the internal tooling that you need to build to manage WordPress at scale?

[00:20:24] BD: Yeah. Well, I think it starts with monitoring. So we have thousands of servers that are running our customers websites. And so we built a significant monitoring stack to kind of observe all the happenings, including things like error rates that are happening, performance metrics that we keep an eye on.

And then in order for operations team to then address and manage any problems that come up, there's just a lot of custom scripts that we run. Some of them have been automated. So they automatically respond to certain alerts and take immediate automated sort of recovery actions. Some of them, as they're being developed, the first iterations of those kinds of things are always manually executed with a human looking eyes on what's happening. So that's kind of the scale out on the operation side.

Then there's the support side of things as well. So WP Engine is known for our world-class technical support. So when you have an issue with your site, and you jump in the user portal and request help, you get someone who is there to help you until your problem is resolved. And to make that happen, they have access to a wide variety of tools that give them insight into

what's happening with your site, changes that have been made, all of the sort of configuration. And they can walk you through sort of getting your site back up online. So that's an entire area of investment that we put a lot of pride in our tooling for our support team.

[00:22:05] JM: And as far as programming language selection for building all those internal tools, I don't suppose you use PHP for all that.

[00:22:14] BD: Originally, we did. So the original WP Engine platform was all PHP. And that was really a pragmatic decision when the team was so small that we needed people who could write code for WordPress, but also kind of write code for the backend systems and not have to context switch between programming languages. As we have grown and matured, we are continuously sort of reducing the size of that original code base and migrating it into – Typically speaking, Python and Go are the major languages that we're building our, what we would call, internal tooling, is all built in those things today.

[00:22:57] JM: Are there any particularly difficult distributed systems problems you've had to tackle to scale up and manage so many WordPress installations?

[00:23:08] BD: It's funny. I don't think we've had to tackle those directly. But things like our monitoring system and other things, we're leveraging other technologies that have those problems sort of embedded in them, right? We use influx dB, which has replicated consensus kind of problems in it. I heard your conversation the other day with Sugu from PlanetScale. That is a thing that we have looked at for a database infrastructure. It's not what we use today. But we are always looking for technologies like that that will help us scale out, for sure.

[00:23:46] JM: Are there any – You said metrics was a focus. And I imagine there's a lot of infrastructure you can build around metrics and alerting? What do you do to ensure speed and uptime? Are there any – Is there any specific work you do around raising the bar of performance that you would get out of just a naive WordPress deployment?

[00:24:11] BD: Yeah, for sure. So we have our performance SLOs that our teams that are always responsible for maintaining. And then periodically, we, as a platform organization, meet and decide whether or not we can sort of ratchet up that floor in terms of uptime or ratchet down the ceiling in terms of acceptable response times and continue to drive additional performance.

We have a whole team that's dedicated to just looking for ways to make the platform faster. So that could be additional caching rules that increase our cache hit rate. It could be building specific versions of PHP with certain optimizations enabled. It can be looking at other things like how Memcached is performing. So Memcached WordPress makes use of temporarily cache objects that it would otherwise load from the database. And so we've done some optimizations there.

We've done optimizations on the file system and the way that works. Yeah, there's lots of things that we're sort of working on at various times to keep, like you said, kind of ratcheting performance up. And then, of course, we definitely benefit also from Moore's law, or whatever it's become now. But as new and improved hardware is available to us in the cloud, we're always early adopters of high-performance hardware in the cloud.

[00:25:47] JM: How do you measure downtime for the WordPress installations? And what happens when an instance goes down? How do you respond? How do you remediate that?

[00:25:59] BD: Yes. So we measure downtime in a couple of different ways. For a good chunk of our platform, we have synthetic monitoring. So we use third parties to monitor those sites and ensure that they're online. In addition, we have sort of internal tooling to make sure that not that the sites that are on – Not the sites themselves are online, but all the infrastructure that manages them is online.

In some cases, there's sort of understood patterns of failure that we have automatic remediations for. And in others, the alerting pages a human who goes to investigate, right? And in all cases, we try to get to the root cause of what's happened and either build better alerting, or better remediation tools, or eliminate the failure mode altogether, if we can.

So we've done a lot of work in the area. In particular, on our scale out infrastructure, looking at how the file system performs – Distributed file systems can get into states where either they're read only, or in some cases, unmounted. And so we sort of detect those situations and remediate them automatically. So that's an example of the kind of automatic remediation that we put in place.

[00:27:26] JM: Tell me more about your usage of cloud infrastructure. Are there some particular database services, infrastructure services, monitoring systems, queueing systems? Any particular things you can outline?

[00:27:41] BD: Yeah. So in our backend systems, we use a lot of different cloud technologies. We use managed databases both from AWS and from GCP. We do use AWS's graph database as part of our backup system. So backups sort of have this graph property to them, which a file can exist in more than one backup. But we only need to ensure that one copy is available so we don't have to have a complete backup at each point in time. So we use a graph database for that.

We use a lot of Pub/Sub, kind of Google Cloud Pub/Sub, to notify different systems when state changes. So if a configuration changes, perhaps through user action, or otherwise, other systems that need to respond to that are notified and take action either to maybe to update additional configuration at a third party partner, like we need to change a DNS record or something, or we need to change a configuration setting on a CDN, something like that, we just kind of a Pub/Sub approach to that.

In many cases now, those are services that are orchestrated in Kubernetes. So we've really kind of gone all-in now on Kubernetes, especially on our backend systems. We have an internally managed software delivery platform. So it's got sort of all of the tooling that a team needs to get up and running quickly on Kubernetes, configure all the service identities and API ingress and any sort of queuing that you need to do that kind of stuff. So yeah, those are the major things that we use, I guess, from the cloud technologies. Oh, and I should say, we also use a significant amount of Big Query at Google for logging events across our platform and using that in our monitoring systems.

[00:29:48] JM: Tell me about the operations teams and how you ensure uptime, and what you do for on-call rotations.

[00:30:00] BD: Yeah. So we have an operations team that is staffed to be a 24 by seven team. So there is always engineers available from the operations team to remediate issues on the core hosting platform. For some of the management services or other ancillary services, the engineering teams themselves, the engineering teams that have developed them, are

responsible. And so each of those engineering teams manages their own on-call rotation for those things.

Oftentimes, there's a sort of primary runbook for the operations team. Like, “Hey, if this service is alerting, here's step one and two to go through to try to get things back online.” And if not, then escalate to the on-call for that service.” And we manage all of that on call rotation then in – We use PagerDuty for that. So all of our teams are in there. And rules are configured to – If alerts are not responded to, escalation happens to a secondary on-call, and then ultimately to the manager, the team. And they will go and wake somebody up if need be. But typically, that doesn't happen. Our on-call folks are on the ball. They're very responsible in terms of managing their alerts.

[00:31:25] JM: What do you spend your time building? Since you have the core WordPress product pretty dialed in, what do you build on top of that?

[00:31:37] BD: Yes. So I think there's a lot of things that customers want more and more access to. Some examples of things that we're working on are customer visible monitoring and alerting, developer tools for building on WordPress, including things like our Genesis, which is our theme framework, Local, which is a tool for developers to get WordPress running locally and then deploy into our environment.

I think we're always looking at the user experience, the end user experience, of how do they interact with our platform? How can we make them more successful faster? That's been a huge focus in the last year trying to make sure that people who sign up for our platform – Many of them are coming to it cold and need more help in getting things live. And so just continuing to refine that user journey of get your site online and be able to understand what's going on with it, situational awareness, that kind of stuff.

I mean, we have a lot of teams working on a lot of things. So it's hard to enumerate them all. But there's always a call for kind of new features or functions that are sort of adjacent to the core WordPress hosting product.

[00:32:53] JM: And on the infrastructure side, is there anything you're doing around platform engineering right now that would be interesting to discuss? Maybe around instrumentation or – I don't know. Service mesh? Or policy management? Or anything like that?

[00:33:10] BD: Yes. So a couple of things in this area. I would say an ongoing effort is this migration into Kubernetes and making sure that more and more of our hosting platform can make it there. And a lot of that comes down to sort of customer – Well, at this point, we're in the middle of migrations. But adding sort of additional capabilities that exist in the prior iteration of our platform to make it possible for those customers who use those features to migrate on to the new platform, because not everything is there yet.

I would say, yeah, other things that are going on. There's definitely a focus on core web vitals metrics as a driver for both internal and customer-facing performance monitoring. So rather than relatively straightforward server time metrics, we're building a system for monitoring that from the end user perspective. And so that's additional telemetry. It's additional data stores to make that happen. So big time series databases and lots of user experience work to make that something that customers can use and get insights from.

[00:34:30] JM: Well, is there anything else you'd like to highlight on the WordPress platform that you build, or WP Engine, or the infrastructure side, or on the analytic side? Any other points that come to mind that would be interesting to discuss in the engineering side?

[00:34:46] BD: You know, one the things that I think is interesting and unique about a platform like ours is it sort of goes to this question of what cloud services do you use? A lot of the cloud services that are out there kind of out of the box are designed to scale out one application, or a small number of applications to massive global reach. And a platform like ours has – Rather than one application that we need to have thousands or hundreds of thousands of replicas of, we have 1.5 million unique websites that we're trying to keep online.

And so a lot of the services that you get kind of out of the box from cloud providers don't scale in that along that axis, right? They scale along a replication axis, not a unique instances kind of scale out. Those are the kinds of problems that we're always working on, because the cloud service tooling just isn't there for those. So that can be managing load balancers, or DNS

records, or ingress rules. Those kinds of things typically are not scaled in the clouds the way we need them to be.

[00:36:12] JM: I guess my last question would be, when you look at these other CMS platforms, modern CMS platforms, how do you think WordPress stacks up against them in terms of engineering and flexibility? And why hasn't WordPress been unseated?

[00:36:34] BD: Yeah, that's a great question. I mean, Wordpress, not only is it not been unseated, it continues to take market share from other CMS systems. Clearly, there's a lot of upstarts who are focusing on sort of point problems for specific types of sites or specific sets of engineers. But the WordPress ecosystem has been around so long that so many of the problems or challenges that you run into, or even new ideas as they come into existence for like new web technologies, or whatever, the community is so large that it responds really relatively quickly by creating plugins for this or that feature or integrating with this new SaaS platform that came online that's relevant. The plugin architecture in WordPress is really flexible. And because of that, it's not just the core developers who sort of bear that responsibility for building the entire ecosystem. It is a platform on which developers of all stripes can build websites, but also build integrations and plugins, businesses that are not strictly their websites, right? So that is the thing that really makes WordPress shine in the community, the size of the community, the engagement. And ultimately, it creates a really robust ecosystem and a robust platform.

[00:38:05] JM: Cool. Well, it's been a real pleasure talking to you. And thank you so much for coming on the show, Brandon.

[00:38:10] BD: Thank you. I really enjoyed the conversation. And I look forward to continue to listen to your podcast.

[00:38:16] JM: All right. Thank you. Thanks for being a listener.

[END]