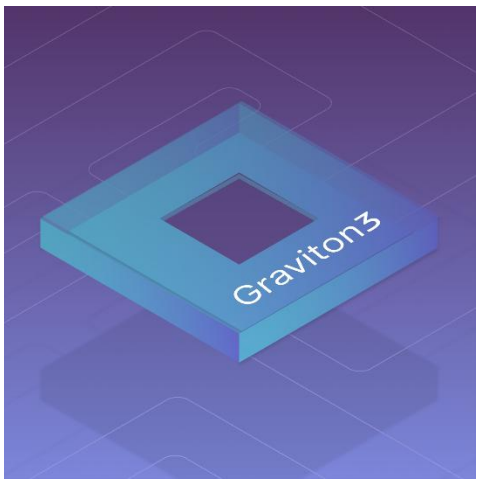
How Braze Embraced AWS Graviton Processors to Reduce Energy Usage and Costs

By Terry Zink, Senior Manager, DevOps, Braze



As a leader in cloud computing, Braze is always looking for new ways to optimize workloads, reduce costs, and increase energy efficiency. When Amazon approached us about AWS Graviton Processors in 2020, we couldn't wait to try them out. But there were several factors that had to be addressed before we could take advantage of this new innovation.

When we began moving forward the following year, Terry Zink, Senior Manager of DevOps, led an initiative involving the DevOps team, and others in the Braze Product, Design, and Engineering (PDE) organization to successfully test using Graviton at Braze. This ensured we could support the new infrastructure and set the stage to roll out the processors across our platform.

For an inside look at how they pulled it off, we sat down with Terry to discuss the challenges, opportunities, and benefits of using Graviton processors to power the Braze product.

As a senior manager of DevOps, I run the cloud operations team. We're responsible for all the compute resources that our service is run on within Amazon Web Services. If you run an application at Braze, it's running on something my team provides 99.9% of the time. So, when Amazon approached us about using Graviton, I knew it would be a great way to reduce the costs and energy required to operate the Braze platform.

What is Graviton?

Unlike traditional servers, which run on either Intel or are AMD-based, Graviton processors are built on a different computer architecture called ARM64. The main advantage of Graviton is that it uses significantly lower power than other processors. That means the costs that come from running computers at scale are significantly reduced, since these processors don't need as much air conditioning power or electricity. Common electronics that run on ARM64 include smartphones, Raspberry Pi, and Apple M1 laptops.

Yet despite the clear advantages, switching to Graviton processors isn’t a simple task. Unlike traditional servers, they don’t use X68 and X64, which is the instruction set that has been common for the past 20+ years. You can’t take an application that runs on what we’re used to and click a button to have it compile on ARM64 instead. And because the technology is new, a lot of software doesn’t support it yet.

Building Infrastructure for Graviton

Braze wanted to take the initiative back in 2020, but the software community wasn’t ready. We rely on a lot of community software, and had to wait for the community to get up to speed before we could run our applications on those platforms as well. For example, Flatcar Linux, the Linux distribution that we run on our Kubra, which is what we run on our Kubernetes, didn’t have support for Graviton until the processors reached Alpha support. And, due to the scale and importance of the processing that happens within the Braze product, we didn’t want to run on Alpha instances, so it wasn’t until mid to late 2021 that we were in a place where we could actively start tackling it.

The official project was to ensure that our Kubernetes platform could support ARM64. We wanted to have the option to run things in our clusters with this, which involved testing to ensure it behaved as expected. That made up about 25% of the work. The actual work required to support ARM64 on our main application, which is our monolithic application on those instances, came as a hack day project. I wasn’t involved personally, but Sal Poliandro, our Senior Director, DevOps Engineering, and Wayne Egerer, Senior Staff Engineer, DevOps, were.

They started by trying to stand up a copy of the platform code base on an ARM-supported version of a Graviton-based host running Linux. Then then

went through the iterative process of chipping away at pieces that failed to run or didn't compile. That included:

- Running into RubyGems that didn't support ARM or wouldn't compile under ARM
- Upgrading the gem stack to the supported version
- Replacing one or two libraries after none of the attempts to run worked right

The next step required us to work with other engineering teams to make sure that this initiative was on their respective roadmaps and prioritized appropriately. Because this project was already on the DevOps team's roadmap, Sal and Wayne's work made it possible to move things forward faster than anticipated, but completing the full project wasn't possible without other key teams buying in and agreeing to put the time in on this effort.

Something in the dashboard required the dashboard team to look into it, and they successfully replaced the library that was causing the issue. Afterward, there was another round of work to package up a proof of concept for this effort. Jason Penny, who's a Senior Software Engineer on the platform infrastructure team, then had to take it and modify all of our build environment to build not only the normal version of the code, but the ARM version as well. Once he did that, we were ready to test it. From an infrastructure perspective, it was designed and built at that point.

Next, it was spun up on a dedicated cluster of machines that are running Graviton instances, and has been running in our staging environment successfully for a couple of months now. We haven't seen any performance issues, and it saves us 20% on average on infrastructure costs. It's also resulted in a huge shift when it comes to our environmental impact, since the energy that it takes to run the Braze product is a significant contributor to our total carbon footprint.

Migrating More Production Infrastructure

Now that we've successfully migrated our core application environments entirely over to Kubernetes, we're able to slowly move more and more of the production infrastructure over to Graviton. Eventually, we'll be running our entire core application on these Graviton instances within Amazon.

We're a relatively early adopter of this new technology, so this is a big win for Amazon and its effort to push for more performant, more energy-efficient cloud infrastructure. It's also huge for our technology stack here at Braze because it means we can expect significant performance gains. Depending on the workload, we can see anywhere from a 2% to 40% increase compared to the old instances, which is great news for the Braze product and for our customers.

However, we do run Braze in a multi-cloud environment, so while we're really excited about the efficiency gains that come with Graviton, we're going to be supporting both systems for the foreseeable future. We don't want to be vendor-locked into a particular cloud, as not all cloud providers support ARM currently. Besides, existing tech stacks aren't going away, and right now the vast, vast majority of all servers on the internet are running on pre-Graviton processors that are dependent on the X68 and X64 instruction set.

Final Thoughts

Overall, this project has been a big success for Braze. While the DevOps team completed 80–90% of the initial work, getting the effort across the finish line and into a place where it could have a real, measurable impact on our systems was a large, multi-team collaboration. And the pieces of the puzzle that the other teams within Braze took on weren't trivial. From creating a shared roadmap to synching backup and handing it off to each consecutive team, everyone had to do their part to get the Graviton processors up and running—and I'm so glad they did.

Braze is hiring for a variety of roles in engineering, product management, and design. Check out our careers page to learn more about our open roles and our culture.