AWS vs Google: flexibility vs operational simplicity

View this email in your browser

This Week In Cloud

Weekly roundup of interesting developments in the cloud industry, curated by David Mytton.

With no interesting announcements last week, I decided to send out a more in depth look at the #1 and #3 cloud providers - AWS and Google. I decided to leave out Azure because I'm less familiar with their offering but I'd place them in a similar category as AWS, at least in the way I compare them to Google below.

AWS vs Google: flexibility vs operational simplicity

Google Cloud Platform and Amazon Web Services are often seen as direct competitors, both offering interchangeable services where one could replace the other e.g. as a user of EC2 you could migrate over to Compute Engine with a very similar set of feature.

This generalisation is true for core services like storage and compute, but AWS have a significant lead shown by the range of products in their portfolio that Google simply does not have. Of course, Google are pushing ahead to close that gap quite rapidly.

However, as Google release their version of different Cloud Platform products, it’s becoming apparent that there is a subtle difference in the approach they are taking with the implementation details which sets them apart from their competitors - operational simplicity.

AWS - a data center as an API

For the most part, AWS products are contained within a single region. As a customer you pick the region you want to operate within and that determines which products are available (rollouts of new releases do not happen to every region at the same time) and pricing, even down to the API endpoints you must use. Although AWS has multiple availability zones within a region, the top level region itself acts like a virtual data centre.

This analogy is useful because it helps to understand how the product features are designed. The most advanced functionality gives you the flexibility and control as if you owned and controlled the data center and all the internal components from the network up. Although there are quick start wizards and sensible defaults, you have the ability to manage things in great detail.

A good example of this is launching a new EC2 instance inside a Virtual Private Cloud, which is now the standard option. You have complete control over all of your networking, from internet gateways to network subnets and from access controls to route tables. This is a true network-as-a-service, managed either through the VPC API or via the AWS

Console. As a result, it's not straightforward and requires some level of understanding, reading of documentation and ongoing management.

This is not a criticism but it does mean there is a shift in mindset when you are coming from other traditional hosting providers where you do not get this level of control. It appears complex, just like managing your own networking hardware might be if you ran your own environment.

Of course, this is the whole idea - AWS has evolved from the original release of basic VMs hiding all the complexity behind the scenes. The goal is to provide these advanced features as if you ran your own networking or compute infrastructure, without actually having to deal with physical hardware, interfaces to complex software, designing and scaling networking infrastructure, and so on.

Google - managing complexity for you

The approach Google has taken to date appears to be focusing on a more subtle simplicity, giving you options but managing most of the complexity.

Compute Engine sustained discounts a good example. These apply automatically as you use an instance throughout the month, with no upfront payment. If you use it, you get the discount. Compare this to AWS's reserved instances where you have different upfront payments, year terms, regions and a marketplace to trade reservations. Google doesn't want you to have to think about this whereas Amazon gives you a range of options and lets you make a decision. In some cases Google is more cost effective, in others AWS wins.

Another example is disk. The performance of Google's block storage depends on the type (persistent disk, SSD or local SSD) and the number of IOPS you can achieve depends on the size of the volume. In contrast, AWS's Elastic Block Storage product has similar types but has different pricing based on the number of IOPS you provision. There's also burst capability which is based on a ticket system so you can't keep bursting. The end product is very similar but the two providers different in how they split out pricing, with AWS giving you more choice.

This philosophy of managed complexity could just be explained by the fact that Google hasn't reached the same level of maturity and feature completeness as Amazon, but you can see the same kind of approach across all the Google Cloud products. Whether it's through removing pricing choices or abstracting locations (Google regions are generically named and they deal with the underlying geographical redundancy rather you having to pick specific locations for your data). The underlying message is that Google only wants you to make the choices that it thinks matter.

It seems like this is the external manifestation of how Google product engineers don't need to care about their own infrastructure for their own products - APIs abstract it all out. Too many choices mean too many chances for mistakes - let the experts deal with it.

Why does this approach matter to users?

Perhaps the clue is in the name. Amazon Web Services are all about providing a public cloud version of services you might have run yourself in the past. This means providing almost the same level of control and functionality as if you had deployed your own hardware in your own data center. AWS users may just accept the defaults but they have a great level of control and flexibility, even if it does add some complexity to managing an AWS environment.

Google Cloud Platform is providing a set of products where many of the decisions are made and managed for you. There is some flexibility and control but the idea of a platform (especially in PaaS) is that the infrastructure is managed for you. You are trading some control and flexibility and instead focusing on the parts of your application that truly matter.

I suspect that as Google continues to develop we will see additional features (and complexity) but this approach of Google managing as much as possible will persist. Amazon, too, will make some improvements (such as the recent simplifying of reserved instances and **removing the need to stripe EBS volumes**) but will continue to offer very advanced functionality as they replace more and more on-premise components. The question is, which do you prefer?

Other things of note

- Google launches an Android management console (**link**)