Comments about the AWS Serverless Workshop

Adrian Kosmaczewski

2022-09-30

Yesterday I attended the AWS Swiss Cloud Day[1] (what a fantastic event!) and participated in the workshop about Serverless computing taught by a very competent team lead by Magdalena Gargas[2] and Scott Gerring[3].

The workshop consisted in the deployment of a quite extensive Node.js application on top of AWS, using not only Lambda[4] but other services, such as Amplify[5] (where the frontend and backend were installed), Cognito[6] (for user management), DynamoDB[7] (database), Cloud9[8] (web-based IDE), CodeCommit[9] (Git repository), and of course IAM[10] (role-based authorizations).

On the positive side, I was glad to see how all of those services come together to create an application! That was very, very interesting, and I thank the AWS team for that. They were also very helpful with all of us, continuously roaming from laptop to laptop to make sure that everyone's code was up and running, and explaining concepts all over the place. A great job indeed.

On the other hand… I hope not to sound too negative, because the presentation was really interesting. But in general I found that the demo app was too big for just one hour and 45 minutes. I am very well aware that AWS has many moving parts, and that it's extremely complicated to distill the essence of such a platform in a few minutes: I have been (I still am, actually) in the position of writing tutorials for technical audiences, and it's really complicated sometimes.

So instead of useless criticism, here's a list of what I would have done differently in such a workshop; of course, I'm no AWS employee nor expert, so please forgive any ignorance on my side.

- I would not have used CodeCommit; I'd simply drop this section. If one can create Lambdas just by pasting code, then let's do that. Cloud9 is needed, though, because of its handy terminal where all the AWS command line tooling is pre-installed.

[1] https://aws.amazon.com/events/swiss-cloud-day/
[2] https://www.linkedin.com/in/magdalenagargas/
[3] https://www.linkedin.com/in/scottgerring/
[4] https://aws.amazon.com/lambda/
[5] https://aws.amazon.com/amplify/
[6] https://aws.amazon.com/cognito/
[7] https://aws.amazon.com/dynamodb/
[8] https://aws.amazon.com/cloud9/
[9] https://aws.amazon.com/codecommit/
[10] https://aws.amazon.com/iam/

- Following the previous point, I'd drop the part of the demo consisting in setting up a static website hosted on Amplify, coupled with user account management part in Cognito. Completing this point and the previous took one hour, and I find that it didn't really add anything to the big picture of "Serverless" computing.
- I'd go directly to the gist of the thing: the famous Lambda function, and, personal taste, instead of using JavaScript for it, I'd use Go instead[11]. Go compiles much faster than JavaScript[12], and it's a very readable and concise programming language; besides, it's also the *de facto* official programming language for the Cloud Native world. The idea is to get a hands-on idea of AWS Lambda as quickly as possible.
- I would have kept DynamoDB just for the Lambda to store data in it; that makes for a useful demo. Read, write, read again, boom, changes!

I always prefer to make demo apps for workshops as simple as possible; in this case, since Serverless is (at least in my mind) somehow synonym of AWS Lambda[13], and if Lambda is all about running snippets of code on demand, I'd just focus on that. Get the code, paste it in the console, and use `curl` to run the request; save and retrieve data. That's it.

From my (admittedly limited) knowledge of AWS, I guess Cloud9 (for its web-based terminal with pre-installed components), Lambda, DynamoDB, and a bit of IAM would have been enough for such a workshop, and all contact with other parts of the infrastructure should be kept to the minimum. In particular IAM: it would of course be really hard not to use it; after all you need to make your Lambda code able to talk to DynamoDB. At the very least.

And if you would like to show a more complex example, you can do so at the end of the session; like a "walkthrough" for attendees, showing how other pieces of the AWS puzzle fit together for more complex scenarios; and you can also make those apps available on GitHub for those interested.

[11] https://docs.aws.amazon.com/lambda/latest/dg/lambda-golang.html

[12] The Node.js application used for this workshop really took a *long* time to compile; couple that to the time it took to deploy the app to Amplify, and this generated lots of idle times in the workshop.

[13] Or maybe it's just that my perception of Serverless is a bit limited; I give you that. After all there's also this thing called Redshift Serverless[14] in their platform, so maybe there's a lot more to that.