UDC 004.514

SERVERLESS ARCHITECTURE

RAMAN KHRAPAVITSKI, DMITRY GLUKHOV
Polotsk State University, Belarus

This article is about serverless architecture using AWS services as an example.

Introduction. Deployment a software application on the Internet typically involves managing some kind of server infrastructure. This typically means the virtual or physical server that you want to manage, and the operating system and other web server hosting processes that are required to run the application. Using a virtual server from a cloud provider such as Amazon means Troubleshooting physical hardware, but still requires a certain level of management of the web server's operating system and software processes. Theserverless architecture [1] focuses on individual functions in the application code. Services such as AWS Lambda [3] functions provide management of all physical hardware, virtual machine operating system, and web server software. You only need to worry about your code.

Main part. Serverless – peer-to-peer application architecture. The architecture is based on microservices, or functions (lambda) that perform a specific task and run on logical containers hidden from prying eyes. That is, the end user is given only the interface for loading the function (service) code and the ability to connect event sources (events) to this function. Using serverless architecture will significantly help developers focus more on their core product. If not for serverless, developers would still be worrying about managing and operating servers or runtimes, whether managing them on the cloud or on-site. This way, the developer's focus will solely be on individual functions in their application code.

Considering the example of Amazon service, the source of events can be many of the Amazon services:

1. S3 storage - generate many events on almost any operation, such as adding, deleting, and editing files in buckets.
2. DynamoDB - allows to generate events on adding or changing data in a table.
3. Cloudwatch is a system similar to cron.
4. API Gateway [4] - software emulator of the HTTP Protocol that allows you to abstract the queries to a single event micro service.

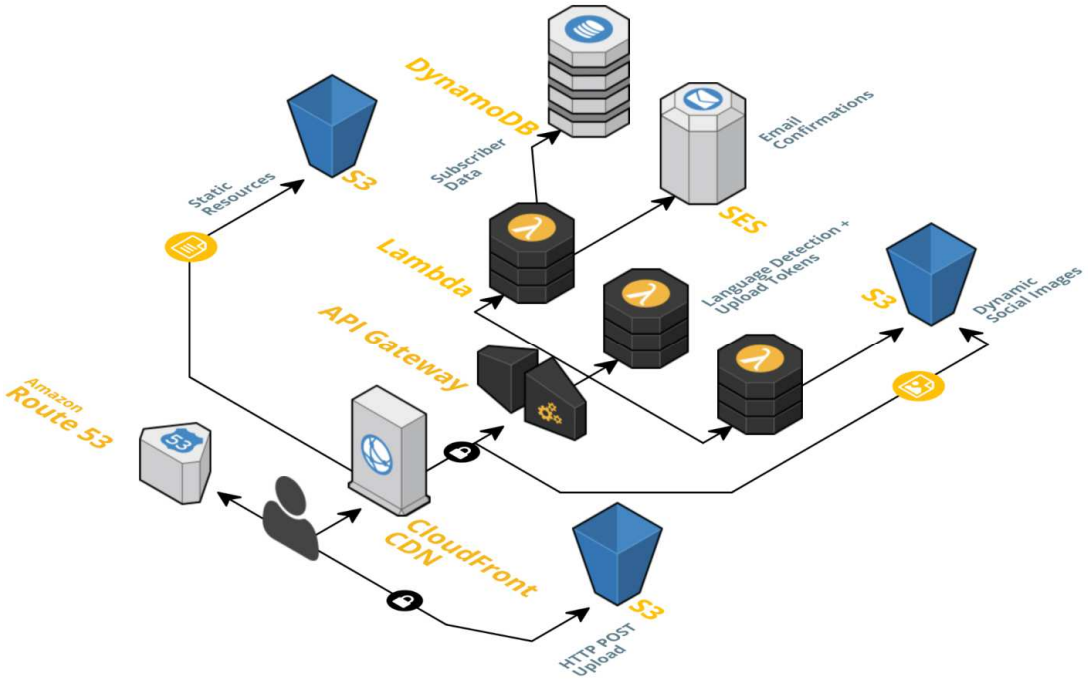

Figure. – General architecture structure

When you load the function code into Amazon, it is stored as a package on an internal file server (like S3). When the first event is received, Amazon automatically runs a mini-container with a specific interpreter (or virtual machine, in the case of Java) and runs the resulting code, substituting the generated event body as an argument. As it is clear from the principle of microservices, each such function can not have a state , since there is no access to the container, and its lifetime is not defined. Due to this quality, microservices can grow horizontally depending on the number of requests and load. In fact, based on the practice, balancing resources in the Amazon performed well and the function is quickly replicated even if the load is not stable.

On the other hand, another advantage of such a stateless startup is that payment for the use of the service can usually be made based on the execution time of a particular function. Such a convenient payment method makes it possible to launch startups or other projects without initial capital. After all, there is no need to buy out hosting for code placement. Payment can be made in proportion to the use of the service (which also allows you to calculate the necessary monetization of your service).

The advantages of this architecture include the following:

1. The lack of a hardware part – servers;
2. Lack of direct contact and administration of the server part;
3. Virtually limitless horizontal growth of your project;
4. Payment for used CPU time.

The disadvantages include:

1. Lack of clear container control (you never know where and how they run, who has access) - which can often cause paranoia.
2. The lack of" integrity " of the application: each function is an independent object, which often leads to a certain dispersion of the application and difficulties to put everything together.
3. The cold start of the container leaves much to be desired (at least in the Amazon). The first launch of a container with a lambda function can often slow down for 2-3 seconds, which is not always well perceived by users.

Conclusion. Serverless technology, in practice, the range of applications of this technology is almost limitless. From simple portals (executed as a static page using React or Angular) with backend and logic on lambda functions to processing archives or files through S3 storage and quite complex mathematical operations with load distribution.

REFERENCES

1. AWS Serverless – AWS Documentation [Электронныйресурс] / AWS Serveless. – Режим доступа: https://aws.amazon.com/serverless/. – Датадоступа: 20.02.2019.
2. AWS Best practices – AWS Documentation [Электронныйресурс] / AWS Architecture best practices. – Режимдоступа: https://aws.amazon.com/architecture/. – Датадоступа: 20.02.2019.
3. AWS Lambda – AWS Documentation [Электронныйресурс] / AWS Lambda. – Режим доступа: https://aws.amazon.com/lambda/. – Датадоступа: 20.02.2019.
4. AWS Api-Gateway – AWS Documentation [Электронныйресурс] / AWS Api-Gateway. – Режим доступа: https://aws.amazon.com/api-gateway/. – Дата доступа: 20.02.2019.