WIKIPEDIA

Serverless computing

Serverless computing is a cloud computing execution model in which the cloud provider runs the server, and dynamically manages the allocation of machine resources. Pricing is based on the actual amount of resources consumed by an application, rather than on pre-purchased units of capacity.[1] It can be a form of utility computing.

Serverless computing can simplify the process of deploying code into production. Scaling, capacity planning and maintenance operations may be hidden from the developer or operator. Serverless code can be used in conjunction with code deployed in traditional styles, such as microservices. Alternatively, applications can be written to be purely serverless and use no provisioned servers at all.[2]

This should not be confused with computing or networking models that do not require an actual server to function, such as peer-to-peer (P2P).

Contents

Serverless runtimes

Most, but not all, serverless vendors offer compute runtimes, also known as function as a service (FaaS) platforms, which execute application logic but do not store data. The first "pay as you go" code execution platform was Zimki, released in 2006, but it was not commercially successful.[3] In 2008, Google released Google App Engine, which featured metered billing for applications that used a custom Python framework, but could not execute arbitrary code.[4] PiCloud, released in 2010, offered FaaS support for Python.[5]

AWS Lambda, introduced by Amazon in 2014,[6] was the first public cloud infrastructure vendor with an abstract serverless computing offering. It is supported by a number of additional AWS serverless tools such as AWS Serverless Application Model (AWS SAM) Amazon CloudWatch, and others.

Google Cloud Platform offers Google Cloud Functions since 2016.[7]

IBM offers IBM Cloud Functions in the public IBM Cloud since 2016.[8]

Microsoft Azure offers Azure Functions, offered both in the Azure public cloud or on-premises via Azure Stack.[9]

Serverless databases

Several serverless databases have emerged in the last few years. These systems extend the serverless execution model to the RDBMS, eliminating the need to provision or scale virtualized or physical database hardware.

Amazon Aurora offers a serverless version of its databases, based on MySQL and PostgreSQL, providing on-demand, auto-scaling configurations. [10]

Azure Data Lake is a highly scalable data storage and analytics service. The service is hosted in Azure, Microsoft's public cloud. Azure Data Lake Analytics provides a distributed infrastructure that can dynamically allocate or de-allocate resources so customers pay for only the services they use.

Google Cloud Datastore is an eventually-consistent document store. It offers the database component of Google App Engine as a standalone service. Firebase, also owned by Google,[11] includes a hierarchical database and is available via fixed and pay-as-you-go plans.[12]

Advantages

Cost

Serverless can be more cost-effective than renting or purchasing a fixed quantity of servers,[13] which generally involves significant periods of underutilization or idle time.[1] It can even be more cost-efficient than provisioning an autoscaling group, due to more efficient bin-packing of the underlying machine resources.

This can be described as pay-as-you-go computing[13] or bare-code[13] as you are charged based solely upon the time and memory allocated to run your code; without associated fees for idle time.[13]

Immediate cost benefits are related to the lack of operating systems costs, including: licences, installation, dependencies, maintenance, support, and patching.[13]

Elasticity versus scalability

In addition, a serverless architecture means that developers and operators do not need to spend time setting up and tuning autoscaling policies or systems; the cloud provider is responsible for scaling the capacity to the demand.[1][9][13]. As Google puts it: ‘from prototype to production to planet-scale.’[13]

As cloud native systems inherently scale down as well as up, these systems are known as elastic rather than scalable.

Small teams of developers are able to run code themselves without the dependence upon teams of infrastructure and support engineers; more developers are becoming DevOps skilled and distinctions between being a software developer or hardware engineer are blurring.[13]

Productivity

With function as a service, the units of code exposed to the outside world are simple event driven functions. This means that typically, the programmer does not have to worry about multithreading or directly handling HTTP requests in their code, simplifying the task of back-end software development.

Disadvantages

Performance

Infrequently-used serverless code may suffer from greater response latency than code that is continuously running on a dedicated server, virtual machine, or container. This is because, unlike with autoscaling, the cloud provider typically "spins down" the serverless code completely when not in use. This means that if the runtime (for example, the Java runtime) requires a significant amount of time to start up, it will create additional latency.

Resource limits

Serverless computing is not suited to some computing workloads, such as high-performance computing, because of the resource limits imposed by cloud providers, and also because it would likely be cheaper to bulk-provision the number of servers believed to be required at any given point in time.

Monitoring and debugging

Diagnosing performance or excessive resource usage problems with serverless code may be more difficult than with traditional server code, because although entire functions can be timed,[2] there is typically no ability to dig into more detail by attaching profilers, debuggers or APM tools. Furthermore, the environment in which the code runs is typically not open source, so its performance characteristics cannot be precisely replicated in a local environment.

Security

Serverless is sometimes mistakenly considered as more secure than traditional architectures. While this is true to some extent because OS vulnerabilities are taken care of by the cloud provider, the total attack surface is significantly larger as there are many more components to the application compared to traditional architectures and each component is an entry point to the serverless application. Moreover, the security solutions customers used to have to protect their cloud workloads become irrelevant as customers cannot control and install anything on the endpoint and network level such as an intrusion detection/prevention system (IDS/IPS). [14]

This is intensified by the mono-culture properties of the entire server network. (A single flaw can be applied globally.) According to protego, the "solution to secure serverless apps is close partnership between developers, DevOps, and AppSec, also known as DevSecOps. Find the balance where developers don't own security, but they aren't absolved from responsibility either. Take steps to make it everyone's problem. Create cross-functional teams and work towards tight integration between security specialists and development teams. Collaborate so your organization can resolve security risks at the speed of serverless."[15]

Privacy

Many serverless function environments are based on proprietary public cloud environments. Here, some privacy implications have to be considered, such as shared resources and access by external employees. However, serverless computing can also be done on private cloud environment or even on-premises, using for example the Kubernetes platform. This gives companies full control over privacy mechanisms, just as with hosting in traditional server setups.

Standards

Serverless computing is covered by International Data Center Authority (IDCA) in their Framework AE360. However, the part related to portability can be an issue when moving business logic from one public cloud to another for which the Docker solution was created. Cloud Native Computing Foundation (CNCF) is also working on developing a specification with Oracle.[16]

Vendor lock-in

Serverless computing is provided as a third-party service. Applications and software that run in the serverless environment are by default locked to a specific cloud vendor.[17] Therefore, serverless can cause multiple issues during migration.[18]

See also

- AWS Lambda
- Cloud Computing
- Function as a service

References

1. Miller, Ron (24 Nov 2015). "AWS Lambda Makes Serverless Applications A Reality" (https://techcrunch.com/2015/11/24/aws-lamda-makes-serverless-applications-a-reality/). *TechCrunch*. Retrieved 10 July 2016.
2. MSV, Janakiram (16 July 2015). "PaaS Vendors, Watch Out! Amazon Is All Set To Disrupt the Market" (https://www.forbes.com/sites/janakirammsv/2015/07/16/paas-vendors-watch-out-amazon-is-all-set-to-disrupt-the-market/). Retrieved 10 July 2016.
3. Williams, Christopher. "Fotango to smother Zimki on Christmas Eve" (https://www.theregister.co.uk/2007/09/25/zimki_fotango_shut/). Retrieved 2017-06-11.
4. "Python Runtime Environment | App Engine standard environment for Python | Google Cloud Platform" (https://code.google.com/appengine/docs/python/runtime.html). *Google Cloud Platform*. Retrieved 2017-06-11.

5. "PiCloud Launches Serverless Computing Platform To The Public" (https://techcrunch.com/2010/07/19/picloud-launches-serverless-computing-platform-to-the-public/). *TechCrunch*. Retrieved 2018-12-17.
6. Miller, Ron (13 Nov 2014). "Amazon Launches Lambda, An Event-Driven Compute Service" (https://techcrunch.com/2014/11/13/amazon-launches-lambda-an-event-driven-compute-service/). *TechCrunch*. Retrieved 10 July 2016.
7. Novet, Jordan (9 February 2016). "Google has quietly launched its answer to AWS Lambda" (https://venturebeat.com/2016/02/09/google-has-quietly-launched-its-answer-to-aws-lambda/). *VentureBeat*. Retrieved 10 July 2016.
8. Zimmerman, Mike (23 February 2016). "IBM Unveils Fast, Open Alternative to Event-Driven Programming" (http://www-03.ibm.com/press/us/en/pressrelease/49158.wss).
9. Miller, Ron (31 March 2016). "Microsoft answers AWS Lambda's event-triggered serverless apps with Azure Functions" (https://techcrunch.com/2016/03/31/microsoft-answers-aws-lambdas-event-triggered-serverless-apps-with-azure-functions/). *TechCrunch*. Retrieved 10 July 2016.
10. "Amazon Aurora Serverless - On-demand, Auto-scaling Relational Database - AWS" (https://aws.amazon.com/rds/aurora/serverless/). *Amazon Web Services, Inc*. Retrieved 2019-08-08.
11. Lardinois, Frederic. "Google Acquires Firebase To Help Developers Build Better Real-Time Apps | TechCrunch" (https://techcrunch.com/2014/10/21/google-acquires-firebase-to-help-developers-build-better-realtime-apps/). Retrieved 2017-06-11.
12. Darrow, Barb (2013-06-20). "Firebase gets $5.6M to launch its paid product and fire up its base" (https://gigaom.com/2013/06/20/firebase-gets-5-6m-to-launch-its-paid-product-and-fire-up-its-base/). *gigaom.com*. Retrieved 2017-06-11.
13. Jamieson, Frazer (4 September 2017). "Losing the server? Everybody is talking about serverless architecture" (http://www.bcs.org/content/conWebDoc/58491).
14. https://www.puresec.io/serverless-security-top-12-csa-puresec
15. Solow, Hillel (2019-02-05). "Serverless Computing Security Risks & Challenges" (https://www.protego.io/serverless-computing-security-risks-challenges/). *protego.io*. Retrieved 2019-03-20.
16. "CNCF, Oracle Boost Serverless Standardization Efforts" (https://www.sdxcentral.com/articles/news/cncf-oracle-boost-serverless-standardization-efforts/2018/05/). *SDxCentral*. Retrieved 2018-11-24.
17. Bashir, Faizan (2018-05-28). "What is Serverless Architecture? What are its Pros and Cons?" (https://hackernoon.com/what-is-serverless-architecture-what-are-its-pros-and-cons-cc4b804022e9). *Hacker Noon*. Retrieved 2019-04-03.
18. "What Is Serverless? Here's a Plain Answer!" (https://squadex.com/insights/what-is-serverless/). *Squadex*. 2019-01-17. Retrieved 2019-04-03.

Further reading

- Roberts, Mike (25 July 2016). "Serverless Architectures" (http://martinfowler.com/articles/serverless.html). *MartinFowler.com*. Retrieved 30 July 2016.
- Jamieson, Frazer (4 September 2017). "Losing the server? Everybody is talking about serverless architecture" (http://www.bcs.org/content/conWebDoc/58491). *BCS, the Chartered Institute for IT*. Retrieved 7 November 2017.

Foundation, Inc., a non-profit organization.