УДК 004.415.2

А. К. Сопрачев (2 курс магистратуры),  
П. Д. Дробинцев., к.т.н., доцент

ПРЕИМУЩЕСТВА ПРИМЕНЕНИЯ SERVERLESS-ПОДХОДА ПРИ РАЗРАБОТКЕ МИКРОСЕРВИСНЫХ ПРИЛОЖЕНИЙ

В современной практике разработки backend-приложений общепринятым подходом к решению задач географического и вычислительного (горизонтального) масштабирования является микросервисная архитектура. Для оркестрации и управления микросервисами практически во всех случаях используют Kubernetes. В данной работе рассматривается альтернативный подход к оркестрации микросервисов — Serverless, а также оцениваются его преимущества и недостатки.

В основе Serverless подхода лежит концепция, при которой разработчику не нужно управлять инфраструктурой на уровне серверов. Вместо этого используются облачные сервисы, предоставляющие в аренду вычислительные ресурсы (объём оперативной памяти, процессорное время, объём хранилища и количество запросов) по модели оплаты по фактическому использованию (pay-as-you-go).

В отличие от традиционного подхода Kubernetes, где необходимо оплачивать выделенные мощности кластера независимо от фактической нагрузки, основным преимуществом Serverless является возможность сокращения расходов за счёт оплаты фактически используемых мощностей без потери автоматического масштабирования. Таким образом необходимо оплачивать только те серверные ресурсы, которые действительно были использованы приложением, что эффективно при переменной нагрузке. Кроме того, этот подход "масштабируется в ноль", что означает полное отсутствие потребления ресурсов при отсутствии нагрузки.

Другим значимым преимуществом является отсутствие необходимости в администрировании и обслуживании серверов, что позволяет разработчикам концентрироваться на создании и улучшении функционала.

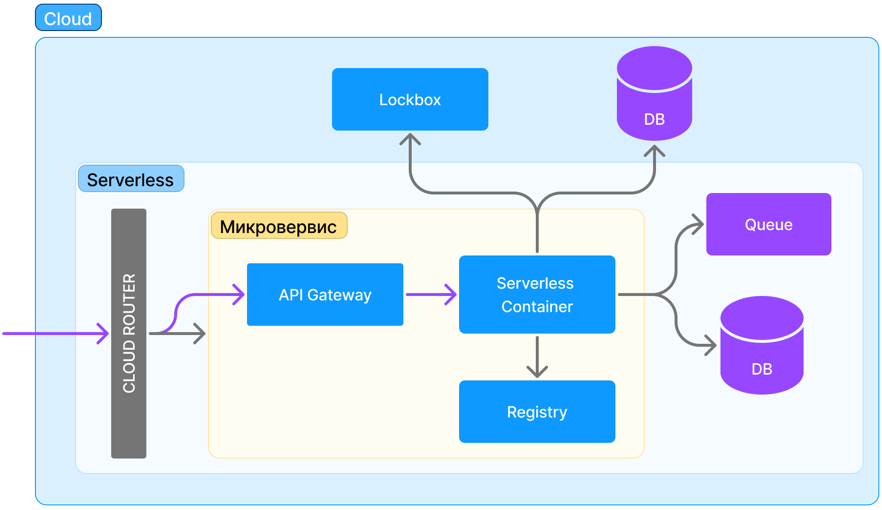


Рис 1. Взаимодействие Serverless сервисов

Реализуется этот подход с помощью использования Serverless сервисов, общие взаимосвязи которых представлены на Рис. 1. Основными сервисами являются:

– API Gateway – внешний интерфейс приложения, описанный по спецификации OpenAPI и являющийся точкой входа.

– Serverless Containers – запускает Docker контейнеры в бессерверном режиме. На каждый запрос создаётся новый экземпляр контейнера, что обеспечивает горизонтальное масштабирование. Тарификация за время использования процессора и оперативной памяти, с квантизацией по 10мс.

– Serverless Database – бессерверные, обычно документоориентированные базы данных с оплатой за использование дискового пространства и количество обращений (например, DynamoDB).

– Object Storage – бессерверное хранилище объектов по протоколу S3, тарификация происходит по объёму данных и числу обращений к ним.

При этом стоит отметить, что Serverless-подход требует строгой архитектурной дисциплины, в частности минимизации зависимостей и соблюдения принципов чистой архитектуры для обеспечения лёгкого масштабирования и независимости компонентов. Также проблемой может стать ограничение облачных провайдеров, которые реализуют не полный перечень сервисов. Однако, применение Serverless подхода не исключает использование классических (не Serverless) сервисов.

Для преодоления сложности ручного конфигурирования облачной инфраструктуры широко применяется подход Infrastructure as Code с использованием Terraform. Этот инструмент позволяет декларативно описывать инфраструктуру в виде конфигурационных файлов, которые можно хранить вместе с кодом микросервисов в системе контроля версий. Совмещение Terraform с Serverless-архитектурой позволяет одной командой разворачивать и обновлять не только production- и dev-окружения, но и создавать новую копию backend на каждый PullRequest, что крайне полезно для проведения End-to-End тестирования.

Таким образом, применение Serverless-технологий в микросервисной архитектуре позволяет эффективно использовать ресурсы, значительно снизить затраты на инфраструктуру и упростить её администрирование.

ЛИТЕРАТУРА

1. Трушин Д. С., Калашников А. О. Применение Terraform для автоматизации управления облачной инфраструктурой // Вестник МГТУ им. Н. Э. Баумана. Серия: Информатика, вычислительная техника и управление. – 2021. – № 4 (142). – С. 65–81.

2. Brisals, S. Serverless Development on AWS: Building Enterprise-Scale Serverless Solutions. O'Reilly Media. – 2020.

3. Jonas E., Schleier-Smith J., Sreekanti V. Cloud Programming Simplified: A Berkeley View on Serverless Computing // arXiv preprint arXiv:1902.03383. – 2019.

4. Kelly, D., Glavin, F. G., & Barrett, E. Denial of wallet – Defining a looming threat to serverless computing. Journal of Information Security and Applications, 58. – 2021

5. Yandex Cloud Serverless. [Электронный ресурс] Режим доступа: https://yandex.cloud/ru/solutions/serverless

6. Terraform by HashiCorp. [Электронный ресурс] Режим доступа: https://www.terraform.io