SAND: Towards High-Performance Serverless Computing

文献报告

1. 关于论文

论文是由 Istemi Ekin Akkus, Ruichuan Chen, Ivica Rimac 等人发表于 USENIX ATC '18。

他们分析了现有无服务计算平台的概况，提出了影响无服务计算性能的两个关键点，针对这两个缺陷，作者提出了并实现了一个新的无服务计算平台：SAND。并比较了同现有无服务平台的运行性能。

2.关于 Serverless Computing

无服务器计算（Serverless Computing）是一种新的云计算架构。它能够使应用程序开发人员不必关心其后端应用程序的基础管理，如服务器管理和容量规划等问题。

要了解 Serverless Computing 就必须从云计算的发展谈起。

在 2006 年，AWS 推出了 EC2（Elastic Compute Cloud），标志着第一代 IaaS（Infrastructure as a Service）的出现。这种服务从本质上讲即将服务器租赁出去并提供基础设施外包服务，用户可以从平台上申请到资源来部署自己的互联网服务。

而后 Heroku 的出现给云计算带来了一个新的服务 PaaS（Platform as a Service）。它给用户提供了一个平台，允许用户在其平台上开发，管理和部署应用程序而不需要关注云端基础架构。

之后便是 serverless 的出现，serverless 一词最早是在 2012 年 Ken Form 所写的一篇名为《Why The Future of Software and Apps is Serverless》的文章出现的。而这篇文章讲的是关于持续集成及源代码控制等内容，并不是我们今天在云计算中的这种架构。

Serverless 在云计算中最开始只有 BaaS（Backend as a Service）这种方式，这种方式是指用于向 web 应用程序和移动应用程序开发人员提供将应用程序链接到后端云存储和后端应用程序公开的 API 的方法，同时还提供诸如用户管理之类的特性推送通知，以及与社交网络服务的集成。

而后随着 2014 年 AWS Lambda 的出现，为无服务器模式带来了一种新的方

式，FaaS（Functions as a Service）。它意味着开发者实现的应用程序逻辑将运行在无状态的计算容器中，它们由事件触发，完全由第三方管理。以 AWS Lambda 为例，它可以允许开发人员直接运行代码而不用配置或管理服务器，只需上传代码，Lambda 就会负责运行所需的一切，并以高可用性拓展代码。开发人员只需设置代码的触发和之间的逻辑关系。

与传统框架相比，FaaS 的优势如下。使用 FaaS 产品，开发人员可以不部署任底层基础架构，使用自己熟悉的语言来创建一个事件触发器。FaaS 还有着很高的拓展性，在负载突然增加时，可以拓展计算容器。而这些完全是自动的，由提供者管理。假如一个应用程序需要并行处理多个请求，那么服务的提供者将负责分配资源来处理这些请求而不需要用户进行任何额外配置，即服务提供者管理底层资源配置和分配，用户只需保证函数的拓展并行性。而这种更加细粒度的服务也降低了成本问题。若果有一个每一分钟运行 5s 的程序，在传统框架中，你需要租赁一个服务器来保证应用程序的运行，即使在大部分的时间里它都是闲置的，而 serverless 的出现是平台提供者可以按照函数运行时间来计费。而开发人员对代码的优化也会提高应用程序的速度，从而直接降低运营成本。

3.面临的问题

首先是延迟的问题，采用 FaaS 的平台在执行处理每个事件之前都需要初始化函数实例，而这将会带来一个几毫秒到几秒之间的延迟。这个延迟取决于很多因素：开发人员所使用的语言，函数所依赖的库的数量，函数的代码量以及平台本身环境的配置等，这种延迟导致了现有的无服务器平台在运行具有多个函数的应用程序时会具有很高的延迟。而现有的无服务器平台通常使用一个全局消息总线来传递事件消息，这种方法会导致一个完整的端到端的函数调用流程，从而引起我们不希望的函数交互延迟，同样的，在一个应用程序逻辑包含多个函数的执行时，这会造成额外的延迟。

其次是函数的执行期限，在 FaaS 框架下，在处理每一个事件消息时都会创建一个函数实例，这个实例的运行时间就是它的执行期限。在 AWS Lambda 中，函数的执行期限最多为 5 分钟。这意味着某些类别的长期任务不适合没有重新架构的 FaaS 平台 – 开发人员可能需要创建几个不同的相互协调的 FaaS 功能，而在传统环境中，可能只需一个长期任务执行协调和执行。

4.现有的平台

首先应该提到 AWS Lambda，正是它的出现给 Serverless Computing 带来了 FaaS 这种方式。这是亚马逊在 2014 年推出的一款商用的无服务器平台，在解决函数启动延迟上做了些改进，平台会通过重新使用已启动的容器来降低函数的启动延迟。

商业无服务器平台有 IBM Cloud Functions ，Microsoft Azure Functions 和 Google Cloud Functions 等，这些平台作为商用平台可以提供完整的 serverless computing 服务。

除此之外，还有一些开源的平台，例如 Apache OpenWhisk，OpenLambda ，Greengrass 和 OpenFaaS 。其中，OpenWhisk 使用 Docker 容器管理基础架构，服务器和扩展。在减少延迟的方式上，OpenWhisk 甚至可以通过提前启动容器来减少函数的启动延迟。

5.相关工作

本文的作者提出了一个新型的无服务器平台，SAND。该平台通过应用程序级的隔离以及分层消息总线来减少函数的启动延迟和交互延迟。改进了现有无服务器平台的一些缺陷。

在 2016 年的 In 8th USENIX Workshop on Hot Topics in Cloud Computing 上，Scott Hendrickson, Stephen Sturdevant 等人在观察了 AWS Lambda 中存在的问题之后提出了 OpenLambda，旨在优化较长的函数启动延迟与较少的局部性考虑。

Garrett McGrath ; Jared Short ; Stephen Ennis ; Brenden Judson 等人研究了现存的各种无服务器框架的延迟，这对于无服务器应用程序开发十分重要。

在 2016 的 14th USENIX Conference on File and Storage Technologies 上，Tyler Harter 等人开发了一个新的容器基准测试 HelloBench，用于评估 57 种不同的容器化应用程序的启动时间，并使用这个和其他发现来指导 Slacker 的设计。Slacker 识别启动容器时关键的包。 通过优先处理这些包并懒惰地加载其他包，它可以减少容器启动延迟。这种改进将使无服务器计算平台受益，特别是那些冷启动的平台。

在 2017 年的 ICDCSW 上，Garrett McGrath , Paul R. Brenner 开发了一个新的无服务器平台，提出了一种排队机制。该机制包含一个冷容器队列和热容器队列，

存在一些 worker 来分配容器和启动容器。通过这种机制提升平台的性能。

6.总结

无服务器计算提供强大的，事件驱动的集成，具有众多云服务，简单的编程和部署模型，以及细粒度的扩展和成本管理。这篇论文讲述了一个新型的无服务器计算平台，作者介绍了平台的设计与实现以及他们在该平台上构建应用程序的经验。采用了应用程序级沙盒以及分层消息总线显著减少了函数启动以及交互延迟。

7.参考文献

[1]. AMAZON. AWS Lambda - Serverless Compute. https://aws.amazon.com/lambda/

[2]. Apache OpenWhisk is a serverless, open source cloud platform.http://openwhisk.apache.org/.

[3]. HARTER, T., SALMON, B., LIU, R., ARPACI-DUSSEAU, A. C., AND ARPACI-DUSSEAU, R. H. Slacker: Fast distribution with lazy docker containers. In 14th USENIX Conference on File and Storage Technologies (FAST 16) (2016).

[4]. HENDRICKSON, S., STURDEVANT, S., HARTER, T.,VENKATARAMANI, V., ARPACI-DUSSEAU, A. C., AND ARPACI-DUSSEAU, R. H. Serverless computation with openlambda. In 8th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 16) (2016).

[5]. MCGRATH, G., AND BRENNER, P. R. Serverless computing: Design, implementation, and performance. In 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW) (2017).

[6]. MCGRATH, G., SHORT, J., ENNIS, S., JUDSON, B., AND BRENNER, P. Cloud event programming paradigms: Applications and analysis. In 2016 IEEE 9th International Conference on Cloud Computing (CLOUD) (2016).

[7]. Apache OpenWhisk is a serverless, open source cloud platform. http://openwhisk.apache.org/