Research project

Powerful Serverless: Power Measurement of Serverless Functions

Mathieu Bacou; SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, France
mathieu.bacou@telecom-sudparis.eu
Romain Rouvoy; Univ. Lille, CRIStAL, UMR CNRS 9189, Inria, France
romain.rouvoy@univ-lille.fr

Keywords: power usage, Serverless, Function-as-a-Service, measurements, systems, cloud

1 Context

Cloud, i.e., renting remote computing resources, is the main method for deploying applications at scale. It has eventually reflected on the architecture of the applications: we observe a trend of "cloud native" designs. Focusing on *Serverless*, applications are designed to deployed on *Function-as-a-Service* (FaaS) platforms. It means that their features are served by composing and replicating simple functions, as illustrated in fig. 1.

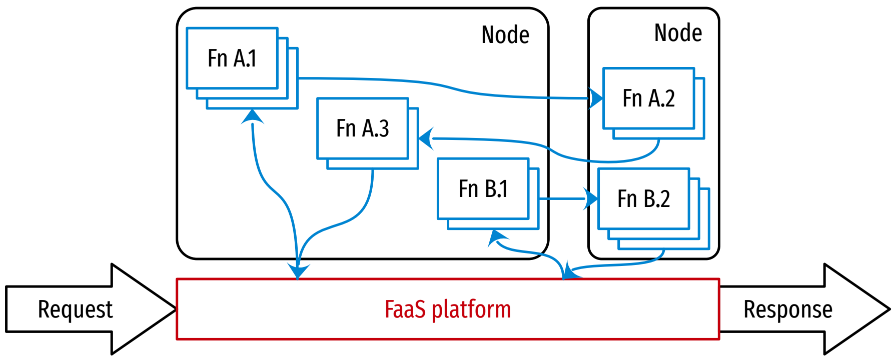

Figure 1: Serving a request with a FaaS application.

It is characterized by the huge number of instances of those functions, the fast rate they are created, used and destroyed, and their quick individual execution time. It makes it hard to understand a critical part of their behavior on the systems: their *power usage*.

Indeed, existing power meter solutions are not fit for the properties of FaaS. Beyond the most basic meters that only measure the global consumption of power of the physical machines, *software-defined power meters* try to give meaningful information about applications. Most often, they rely on some previous training that requires a lot of data and time. This limit makes them largely impractical, especially in FaaS environments, because the requirement is to understand the power usage behavior of user-provided workloads that is by definition unknown at first. To deal with this constraint, although not specifically in FaaS environments, the French Inria team Spirals developed PowerAPI [7]. Citing the companion paper, PowerAPI is "a software toolkit for assembling software power meters, [...] to monitor power consumption of software".

Of interest is PowerAPI's ability to estimate power consumption at different granularity levels: process, thread, container, virtual machine, etc. Indeed, FaaS platforms are built as deep and distributed stacks of software system layers, including operating systems, hypervisor, cluster orchestrators, language runtimes, etc. Furthermore, power models used by PowerAPI are continuously self-calibrated to consider current execution conditions of the machine hosting the software. This is an important feature for FaaS servers, which workloads can vary greatly over short periods of time.

Nonetheless, while a promising base, PowerAPI suffers from design- and system-level limits shared by other power meters that monitor at the container level, and have been designed with micro-services in mind [8, 1, 5]. Those limits make them unfit for the characteristics of FaaS given above. In parallel, the software project FaaSLoad [3, 4] was started to provide a solution to the issues of monitoring serverless functions. FaaSLoad can inject a workload into a serverless platform, and collect data at the level of function instances. However it is currently limited to performance data collected by perf [6] and blktrace [2].

2 Goals

In the Powerful Serverless projet, the goal is to tackle the challenges of *monitoring the power usage* of Serverless functions. By leveraging the existing infrastructure of FaaSLoad, and the software-defined power meter solution of PowerAPI, the proposed work is to explore the research issues linked to:

- adapting power usage measurement to the properties of FaaS: scale of instances to monitor, special life cycle, and quick execution time;
  - how can power usage modeling benefit from the properties of FaaS, what are the impacts of those properties on the classical assumptions about this modeling, ...
- bringing the system layers up to the job of power usage measurement of Serverless functions: high frequency, and high concurrency of metrics collection;
  - do the hardware and the kernel expose the necessary information at the necessary frequency, granularity, with enough scalability, ...

3 Work environment

The student(s) will work under the direct supervision of Mathieu Bacou, maintainer of FaaSLoad, and under the supervision of Romain Rouvoy, a core contributor and director of PowerAPI working in Lille. They will integrate into the research team Benagil of Inria and the Parallel and Distributed Systems (PDS) team of the CS department of Télécom SudParis.

Technology-wise, the student(s) will gain experience in bleeding-edge cloud infrastructures, with FaaS platforms (mainly Apache OpenWhisk), and cluster management (containers, Kubernetes) applied to state-of-the-art issues of the cloud. Development can be expected in Python (both for FaaSLoad and PowerAPI) as well as systems language (C for PowerAPI and work in the Linux kernels, otherwise left to the students' preference).

References

[1] Marcelo Amaral et al. "Process-Based Efficient Power Level Exporter". In: *17th IEEE International Conference on Cloud Computing, CLOUD 2024, Shenzhen, China, July 7-13, 2024*. Ed. by Rong N. Chang et al. IEEE, 2024, pp. 456–467. DOI: 10.1109/CLOUD62652.2024.00058. URL: https://doi.org/10.1109/CLOUD62652.2024.00058.

[2] Jens Axboe. *Block IO Tracing*. Computer Software. 2006. URL: https://git.kernel.org/pub/scm/linux/kernel/git/axboe/blktrace.git/about/.

[3] Mathieu Bacou. "FaaSLoad: Fine-Grained Performance and Resource Measurement for Function-As-a-Service". In: *28th International Conference on Principles of Distributed Systems, OPODIS 2024, December 11-13, 2024, Lucca, Italy*. Ed. by Silvia Bonomi et al. Vol. 324. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, 22:1–22:21. DOI: 10.4230/LIPICS.OPODIS.2024.22. URL: https://doi.org/10.4230/LIPIcs.OPODIS.2024.22.

[4] Mathieu Bacou. *FaaSLoad: Function-as-a-Service workload injector*. Version 2.1.0. Computer Software. 2024. URL: https://gitlab.com/faasload/faasload.

[5] Rolando Brondolin, Tommaso Sardelli, and Marco D. Santambrogio. "DEEP-Mon: Dynamic and Energy Efficient Power Monitoring for Container-Based Infrastructures". In: *2018 IEEE International Parallel and Distributed Processing Symposium Workshops, IPDPS Workshops 2018, Vancouver, BC, Canada, May 21-25, 2018*. IEEE Computer Society, 2018, pp. 676–684. DOI: 10.1109/IPDPSW.2018.00110. URL: https://doi.org/10.1109/IPDPSW.2018.00110.

[6] Linux kernel contributors. *perf: Linux profiling with performance counters*. Computer Software. 2009. URL: https://perf.wiki.kernel.org/index.php/Main_Page.

[7] Guillaume Fieni et al. "PowerAPI: A Python framework for building software-defined power meters". In: *J. Open Source Softw.* 9.98 (2024), p. 6670. DOI: 10.21105/JOSS.06670. URL: https://doi.org/10.21105/joss.06670.

[8] Jan Treibig, Georg Hager, and Gerhard Wellein. "LIKWID: A Lightweight Performance-Oriented Tool Suite for x86 Multicore Environments". In: *39th International Conference on Parallel Processing, ICPP Workshops 2010, San Diego, California, USA, 13-16 September 2010*. Ed. by Wang-Chien Lee and Xin Yuan. IEEE Computer Society, 2010, pp. 207–216. DOI: 10.1109/ICPPW.2010.38. URL: https://doi.org/10.1109/ICPPW.2010.38.