

# Container Start Times: Empirical Analysis and Predictability

Martin Straesser<sup>1</sup>, Samuel Kounev<sup>1</sup>

<sup>1</sup>Universität Würzburg, Sanderring 2, 97070 Würzburg, Germany

## Keywords

Container, Cloud Computing, Empirical Study

In recent years, container technology gained more and more importance, especially for the deployment of cloud services. While many studies focus on the performance characteristics of running containers, only a few explicitly address container start times. However, a good understanding of those can help performance engineers and cloud service providers in various domains. For example, autoscaling and scheduling algorithms can profit from additional knowledge about container start times, if they explicitly consider start delays for different containers. Another related research field is serverless computing, where the optimization of container startups and the analysis and mitigation of cold starts are vital research topics. Therefore, container start times and their influencing factors need to be analyzed in a systematic way.

State-of-the-art literature focuses mainly on comparative analyses of different container runtimes and their influence on the container start time [1, 2]. Only few consider additional influencing factors like storage driver [3], base images [3, 4] as well as particular deployment aspects [5]. Moreover, the majority of these studies have only short evaluations with few container images and measurement repetitions as well as non-standardized methodologies.


In this work, we perform a systematic and extensive study to identify influencing and negligible factors for container start times. The start process of a container begins when the container engine receives the task to start a new container. It ends with the start of the execution of the container's start command defined in the image manifest. The container start time is the time elapsed between these two events. During the design and execution of this study, we explicitly work with the recent ACM SIGSOFT Empirical Standards [6].

We group potential influencing factors for container start times in two categories: image factors and deployment factors. The image factors are all information that is extractable from the container's image manifest. This includes the image size, number and size of layers, ports, volumes, and more. Deployment factors include the node OS, kernel and architecture, container runtime, storage driver as well as other metrics, like the number of running containers on the node. The pull or download time of the container image will not be investigated further, as this is mostly a matter of network conditions.

Our goal is to predict container start times based on image and deployment information. Our study is divided into three stages. In stage one, we will perform extensive measurements of container start times with a large number of images from the prominent public repository


---

SSP'21: Symposium on Software Performance, November 09–10, 2021, Leipzig, Germany

 martin.straesser@uni-wuerzburg.de (M. Straesser); samuel.kounev@uni-wuerzburg.de (S. Kounev)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

Docker Hub. A study analyzing the characteristics of container images on Docker Hub was published in 2019 [7]. By doing this, we obtain a systematic and nearly exhaustive investigation of the image factors as well as statistical characteristics of container start times, like empirical variances. In stage two, we will analyze the results from stage one and focus on deployment factors by changing node configurations and other parameters. After completing stage two, we have a list of relevant factors, which influence container start times. Stage three of this study will be concerned with the predictability of container start times. We want to answer the question: *Can we predict the start time of an arbitrary container given its image manifest and deployment information?* Therefore, we train a prediction model based on the measurements from the previous stages. The obtained results are then verified by images from other public repositories or self-created images.

In this talk, we present the study design, experiment setup, and preliminary results of our study. Moreover, we elaborate more about the motivation and potential use cases. We hope that the discussion will help us to discuss the strengths and weaknesses of this study regarding both the methodological and technical aspects.

## References

- [1] L. Espe, A. Jindal, V. Podolskiy, M. Gerndt, Performance evaluation of container runtimes., in: 10th International Conference on Cloud Computing and Services Science (CLOSER), 2020, pp. 273–281.
- [2] R. Kumar, B. Thangaraju, Performance analysis between runc and kata container runtime, in: 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), 2020, pp. 1–4.
- [3] G. E. de Velp, E. Rivière, R. Sadre, Understanding the performance of container execution environments, in: Proceedings of the 2020 6th International Workshop on Container Technologies and Container Clouds, Association for Computing Machinery, New York, NY, USA, 2020, p. 37–42.
- [4] T. Yoshimura, R. Nakazawa, T. Chiba, Imagejockey: A framework for container performance engineering, in: 2020 IEEE 13th International Conference on Cloud Computing (CLOUD), 2020, pp. 238–247.
- [5] A. Lingayat, R. R. Badre, A. Kumar Gupta, Performance evaluation for deploying docker containers on baremetal and virtual machine, in: 2018 3rd International Conference on Communication and Electronics Systems (ICCES), 2018, pp. 1019–1023.
- [6] ACM SIGSOFT empirical standards for conducting and evaluating research in software engineering, 2020. URL: <https://acmsigsoft.github.io/EmpiricalStandards/docs/>.
- [7] N. Zhao, V. Tarasov, H. Albahar, A. Anwar, L. Rupperecht, D. Skourtis, A. S. Warke, M. Mohamed, A. R. Butt, Large-scale analysis of the docker hub dataset, in: 2019 IEEE International Conference on Cluster Computing (CLUSTER), 2019, pp. 1–10.