# Challenges of PostgreSQL resource sharing

Dmitrii Dolgov[1]

[1]) Zalando SE
9erthalion6@gmail.com

One of the modern trends in the databases world is the migration from a bare metal and managing the data on top of some abstract layer, like virtualization or containerization. In some cases this approach indeed makes sense due to more flexibility (if one wants to build a DBaaS) and scalability or cost reduction. But of course such advantages always comes as a trade off with the performance due to resource sharing between different clients and technical complexity.

Taking this into account and having some reasonable experience with running databases on top of different platforms such as AWS and Kubernetes, we want to show (on PostgreSQL as an example) that:

- Of course it could be challenging to make it right, but contrary to the common opinion e.g. that containers are "no go" area for databases, it's quite possible.
- Having some layers between a database and hardware means that it's not possible anymore to investigate and reason about the performance only from a data storage point of view.

We will discuss the performance overhead, that different abstraction layers (such as VM, container or K8S) incur on PostgreSQL and how to measure and make it visible, exploiting the underlying platform. For that we will explain how resource sharing approach on different layers (e.g. cgroups) affect the database. Some of such effects we have faced on the production systems, some are possible only in rare cases. But what they all have in common is that it's virtually impossible to investigate only from within the PostgreSQL database itself and one needs to deploy strace, perf, eBPF to achieve this (e.g. lock preemption problem on VM, shared memory segments limitations or memory reclaim process within a container).