GNU Taler Scalability

Currently, central banks all over the globe are investigating possible implementations of a Central Bank Digital Currency (CBDC). Unfortunately, most of today's electronic payment systems do either not offer adequate technical privacy assurances to citizens, or are too slow to handle the expected transaction load.

One of the goals of the GNU Taler project is to provide a free software reference implementation for a privacy preserving CBDC.

Introduction

According to a personal discussion with Giesecke+Devrient, a payment system that is to support 500 million people for all payments should be able to handle about 100'000 transactions per second (TPS). This would be sufficient to handle the combined rate of **all** currently used means of payment, such as credit cards, bank transactions and cash. Naturally, a smaller economy might work well on a much lower transaction rate. For example, the same per-capita use would imply a need of 2'000 TPS for all of Switzerland. For comparison, Visa currently achieves an average TPS of 1'677 worldwide.

The goal of this work was to assess the scalability of GNU Taler in a distributed system.

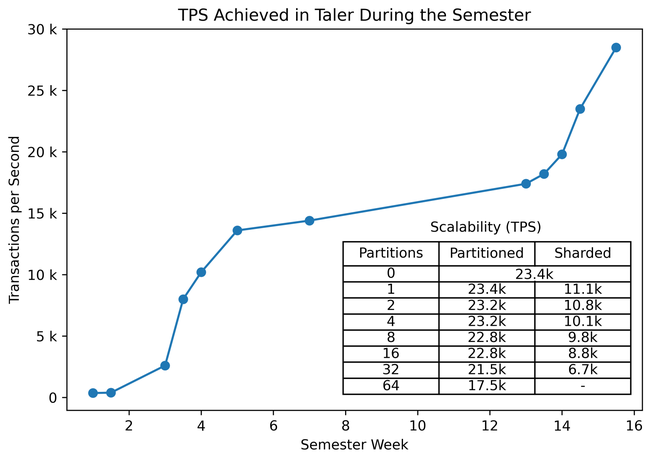
Method

To assess the scalability of GNU Taler, performance experiments were carried out using INRIA's Grid'5000. Grid'5000 is a distributed testbed providing computing and storage resources across France (and Luxembourg). It allows experimenters to reserve groups of machines for experiments and have direct access to the local hardware and communication over shared high-speed network links.

We have created a dynamic and reproducible experimental setup that can be used for further analysis. This includes extracting node metrics with Prometheus, log collection and calculation of statistics with Promtail and Loki, and visualization of the collected data with Grafana dashboards.

Results

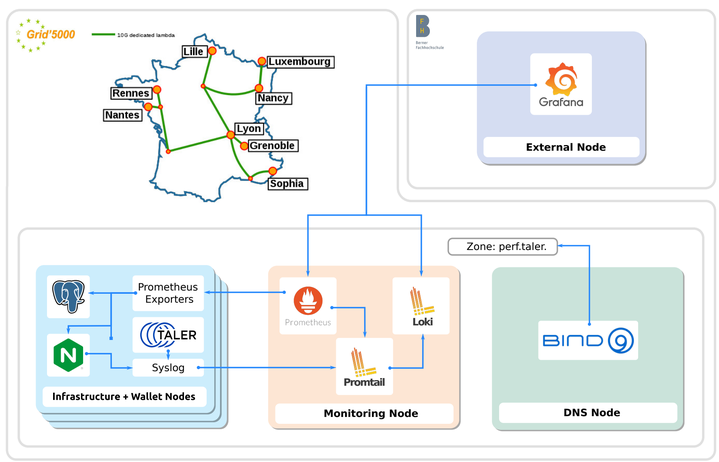
Experiments were conducted with uneven load distribution between merchant accounts to increase realism. We were able to identify and fix several bottlenecks in the GNU Taler system: We parallelized the execution of the cryptographic front-end leaving the PostgreSQL database as the natural bottleneck. Here we had major problems with serialization errors. But these were almost completely eliminated by partitioning the database and appropriately instructing the query planner to match the query execution to these partitions. We demonstrated that Taler scales well in a distributed environment, including horizontal distribution of the PostgreSQL database. Our experiments show that CBDCs based on Taler would only require a few exchanges per continent.



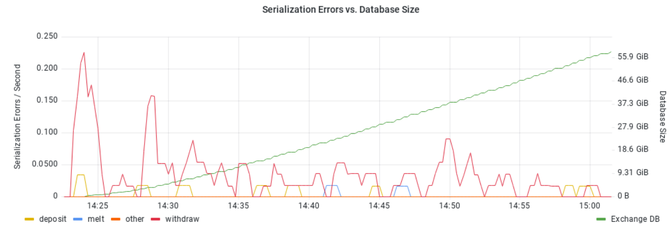
Scalability (TPS)

Partitions	Partitioned	Sharded
0	23.4k	
1	23.4k	11.1k
2	23.2k	10.8k
4	23.2k	10.1k
8	22.8k	9.8k
16	22.8k	8.8k
32	21.5k	6.7k
64	17.5k	-

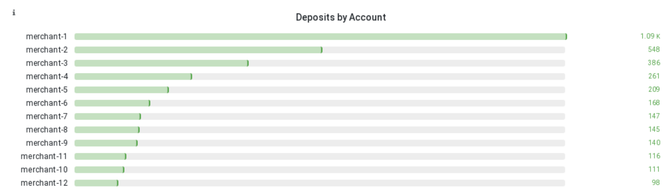
Starting with 300 TPS, we were able to increase the performance of GNU Taler by a factor of 95 up to 28.5k.



Network and experiment system architecture on Grid'5000 and the BFH.



Remaining serialization errors in the PostgreSQL database while a total of 65k transactions are made in the DB.



The Zipf distribution of payments to the different merchant accounts for increased realism.

Bern University of Applied Sciences

Graduate: Boss Marco
Advisor: Prof. Grothoff Christian
Co-Advisor: Prof. Habegger Andreas
Expert: Van der Kleij Han

Degree Program: Computer Science
Specialization: IT-Security