NDBench: Benchmarking Microservices at Scale
(submitted to HPTS 2017)

Ioannis Papapanagiotou
Netflix

Vinay Chella
Netflix

Netflix runs thousands of microservices and thousands of backend data store nodes on Amazon's Web Services (AWS) instances to serve and store data for almost 100M users. Hence, we are not always aware of the traffic that bundled microservices may generate on our backend systems. Understanding the performance implications of new microservices on our backend systems was also a difficult task. We needed a framework that could assist us in determining the behavior of our scalable and highly available persistent storage systems under various workloads, maintenance operations, and capacity constraints. We also wanted to be mindful of provisioning our clusters, scaling them either horizontally (by adding nodes) or vertically (by upgrading the instance types), and operating under different workloads and conditions, such as node failures, network partitions, etc.

As new NoSQL data store systems appear in the market or pluggable database engines - Log-Structured-Merge-Database (LSM) design, B-Tree, Bw-Tree, HB+-Trie etc. - they tend to report performance numbers for the sweet spot, and are usually based on optimized hardware and benchmark configurations. Being a cloud-native enterprise, we want to make sure that our systems can provide high availability under multiple failure scenarios, and that we are utilizing our resources optimally. There are many other factors that affect the performance of a database deployed on the Cloud, such as the instance types, workload patterns, and types of deployments such as island vs global, number of replicas etc. There were also some additional requirements; for example, as we upgrade our data store systems (such as Apache Cassandra upgrades) we wanted to test the systems prior to deploying them in production.

For systems that we develop in-house, such as Dynomite, we wanted to automate the functional test pipelines, understand the performance of Dynomite under various conditions, and under different storage engines, e.g. Redis, RocksDB, LMDB, ForestDB, etc. Hence, we wanted a workload generator that could be integrated into our pipelines for validating production-ready AMIs. Finally, we wanted a tool that would resemble our production deployment. In the world of microservices, a lot of business process automation is driven by orchestrating across services. Traditionally, some of the benchmarks were performed in an ad-hoc manner using a combination of APIs, making direct REST calls in each microservice, or using tools that directly send traffic to the data stores. However, most benchmark tools are not integrated as microservices and do not have out of the box cluster management capabilities and deployment strategies.

Hence, we architected and developed NDBench. Netflix Data Benchmark (NDBench) is a pluggable cloud-enabled benchmarking tool that can be used across any data store system. NDBench aids in simulating the performance benchmark by mimicking several production use cases. NDBench allows for dynamically changing the benchmark configurations while a test is running, hence perform tests along our production microservices. NDBench is well integrated with common platform cloud services such as dynamic configuration management, discovery services, and metrics. NDBench allows us to run infinite horizon tests to identify potential memory leaks from long running processes or garbage collection issues, or to test long running maintenance jobs such as database repairs or reconciliation. NDBench provides pluggable patterns, load configuration, and interfaces for supporting different client APIs. NDBench includes a state-of-the-art UI for deploying, managing and monitoring multiple instances from a single entry point reducing the overhead of the person doing the benchmarks. We also incorporated NDBench into the Netflix Open Source (OSS) ecosystem by integrating it with components such as Archaius for configuration, Spectator for metrics, and Eureka for discovery service, and Spinnaker for global continuous delivery. However, we designed NDBench so that these libraries are injected, allowing the tool to be ported to other cloud environments, or run locally. In this talk, we are going to cover the technology, the use cases and showcase a demo of NDBench. We have recently open sourced the framework.

Short bio of corresponding author

Ioannis Papapanagiotou is an experienced engineer and researcher having served different positions in the industry and academia. He holds a dual Ph.D. degree in Computer Engineering and Operations Research from NC State University. He is currently a senior distributed systems architect at Netflix, a research assistant professor at the University of New Mexico, and a graduate faculty at Purdue University. He focuses on distributed systems, cloud computing, internet of things, and network systems.

Ioannis has served in the faculty ranks of Purdue University (tenure-track), NC State University and the University of New Mexico. He has been awarded the NetApp faculty fellowship and established an Nvidia CUDA Research Center at Purdue University. Ioannis has also received the IBM Ph.D. Fellowship, Academy of Athens Ph.D. Fellowship for his Ph.D. research, and best paper awards in several IEEE conferences for his academic contributions. Ioannis has authored approximately 40 research articles and 10 patent disclosures. Ioannis is a member of ACM and senior member of IEEE.