Gorilla: A Fast, Scalable, In-Memory Time Series Database

Tuomas Pelkonen Scott Franklin Justin Teller
Paul Cavallaro Qi Huang Justin Meza Kaushik Veeraraghavan
Facebook, Inc.
Menlo Park, CA

ABSTRACT

Large-scale internet services aim to remain highly available and responsive in the presence of unexpected failures. Providing this service often requires monitoring and analyzing tens of millions of measurements per second across a large number of systems, and one particularly effective solution is to store and query such measurements in a time series database (TSDB).

A key challenge in the design of TSDBs is how to strike the right balance between efficiency, scalability, and reliability. In this paper we introduce Gorilla, Facebook's in-memory TSDB. Our insight is that users of monitoring systems do not place much emphasis on individual data points but rather on aggregate analysis, and recent data points are of much higher value than older points to quickly detect and diagnose the root cause of an ongoing problem. Gorilla optimizes for remaining highly available for writes and reads, even in the face of failures, at the expense of possibly dropping small amounts of data on the write path. To improve query efficiency, we aggressively leverage compression techniques such as delta-of-delta timestamps and XOR'd floating point values to reduce Gorilla's storage footprint by 10x. This allows us to store Gorilla's data in memory, reducing query latency by 73x and improving query throughput by 14x when compared to a traditional database (HBase)-backed time series data. This performance improvement has unlocked new monitoring and debugging tools, such as time series correlation search and more dense visualization tools. Gorilla also gracefully handles failures from a single-node to entire regions with little to no operational overhead.

1. INTRODUCTION

Large-scale internet services aim to remain highly-available and responsive for their users even in the presence of unexpected failures. As these services have grown to support a global audience, they have scaled beyond a few systems running on hundreds of machines to thousands of individual systems running on many thousands of machines, often across multiple geo-replicated datacenters.

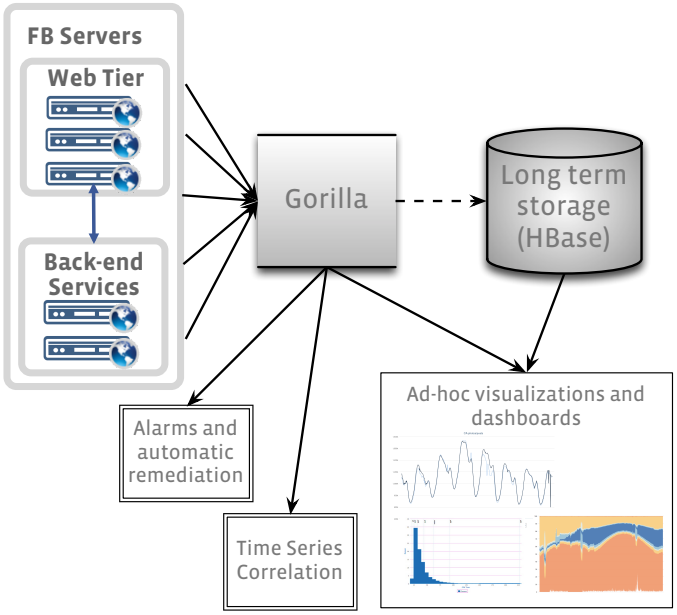

Figure 1: High level overview of the ODS monitoring and alerting system, showing Gorilla as a write-through cache of the most recent 26 hours of time series data.

An important requirement to operating these large scale services is to accurately monitor the health and performance of the underlying system and quickly identify and diagnose problems as they arise. Facebook uses a time series database (TSDB) to store system measuring data points and provides quick query functionalities on top. We next specify some of the constraints that we need to satisy for monitoring and operating Facebook and then describe Gorilla, our new in-memory TSDB that can store tens of millions of datapoints (e.g., CPU load, error rate, latency etc.) every second and respond queries over this data within milliseconds.

Writes dominate. Our primary requirement for a TSDB is that it should always be available to take writes. As we have hundreds of systems exposing multiple data items, the write rate might easily exceed tens of millions of data points each second. In constrast, the read rate is usually a couple orders of magnitude lower as it is primarily from automated systems watching 'important' time series, data

 Articles from this volume were invited to present their results at the 41st International Conference on Very Large Data Bases, August 31st - September 4th 2015, Kohala Coast, Hawaii.
Proceedings of the VLDB Endowment, Vol. 8, No. 12