tsdb: A Compressed Database for Time Series

Luca Deri[1,2], Simone Mainardi[1,3], and Francesco Fusco[4]

[1] Institute of Informatics and Telematics, CNR, Pisa, Italy
[2] ntop, Pisa, Italy
[3] Department of Information Engineering, University of Pisa, Pisa, Italy
[4] IBM Zürich Research Laboratory, Rüschlikon, Switzerland
{luca.deri,simone.mainardi}@iit.cnr.it
ffu@zurich.ibm.com

Abstract. Large-scale network monitoring systems require efficient storage and consolidation of measurement data. Relational databases and popular tools such as the Round-Robin Database show their limitations when handling a large number of time series. This is because data access time greatly increases with the cardinality of data and number of measurements. The result is that monitoring systems are forced to store very few metrics at low frequency in order to grant data access within acceptable time boundaries.

This paper describes a novel compressed time series database named tsdb whose goal is to allow large time series to be stored and consolidated in real-time with limited disk space usage. The validation has demonstrated the advantage of tsdb over traditional approaches, and has shown that tsdb is suitable for handling a large number of time series.

Keywords: Large-scale datasets, time series, web-based data visualization.

1 Introduction and Motivation

The demand of (near) real-time monitoring as well as the analysis of high-speed networks has put several constraints on monitoring systems. Users are demanding solutions able to interactively drill-down data while simultaneously collecting (hundred of) thousand metrics from various network sensors. In order to increase measurement accuracy, network administrators often reduce the sampling frequency of counters and gauges. If some years ago, a sampling period of 5 minutes was acceptable, nowadays network administrators require higher frequency samples for detecting anomalies that would not be detected by monitoring the same data at lower frequencies. For instance, detection of traffic spikes and microbursts require tenth (if not hundred) samples per second. The consequence of this trend is that monitoring systems produce an ever-increasing amount of data that needs to be stored and analyzed in a limited amount of time.

With the advent of multi-Gbit networks, the traffic being analyzed and the corresponding number of measured metrics increased significantly. Periodically accounting host traffic for a /24 subnet is very different from performing the same activity on a network backbone. In the latter case, monitoring systems cannot tolerate delays while

A. Pescapè, L. Salgarelli, and X. Dimitropoulos (Eds.): TMA 2012, LNCS 7189, pp. 143–156, 2012.
© Springer-Verlag Berlin Heidelberg 2012

saving/reading data from/to the disk, as data access slow-down would prevent the system from carrying on the tasks within the expected timeframe.

As discussed in the following section, both relational databases and specialized tools such as the rrdtool [3], are used in the industry for handling *time series*. A time series is a sequence of data points measured at uniform time intervals (e.g. every 5 minutes) [16]. Unfortunately both solutions are only capable of satisfying requirements coming from small to medium environments, where the number of monitored metrics does not exceed a few tenth of thousand. However, collecting a much higher number of time series is not uncommon these days. Even a simple ntop installation [4] deployed for monitoring a medium network, has to keep track of several tens of thousand metrics just for keeping a few counters (e.g. bytes/packets sent/received). If additional counters are measured (e.g. traffic per protocol and network), the dataset size can quickly increase. According to the tests we carried on, existing open-source solutions and relational databases are affected by serious scalability issues when collecting hundreds of thousands (millions) metrics, making them *practically unusable* for large monitoring systems. In fact, without scalable and efficient time series handling tools, monitoring systems cannot store data at fine-grained granularity and are forced to decrease the monitoring accuracy. The lack of open-source tools for an efficient handling of time series has been the main motivation for creating a new open-source time series database for time series called *tsdb*.

The rest of the paper is organized as follows. Section 2 analyzes the various alternatives for handling time series. Section 3 presents the design and implementation of tsdb. Section 4 covers the tsdb validation and compares it with similar tools. Section 5 highlights some open issues and future work items.

2 Background and Related Work

2.1 Database Systems

For years network developers have used relational databases for storing network data persistently. Data with uniform characteristics are organized in tables linked by relationships. Each table is logically divided in columns and rows, where a row is uniquely identified by means of a primary key. Data stored on the database can be modified and deleted. Unfortunately network data is not characterized by many relationships, it is usually unchangeable (i.e. changing measured data might indicate a counterfeit), and the same data is repeated over time at every measurement interval, making relational databases not convenient for handling this type of data.

The reasons are manifold:

- At every measurement interval, tables are populated with fresh data that increases table cardinality. The consequence is that table cardinality as well the space taken on disk increases with the number of measurements.
- As soon as table indexes become large enough to prevent themselves to be cached, data retrieval becomes significantly slow [21, 14] thus jeopardizing the performance of applications sitting on top of the database.