A Distributed Network Fluctuation Monitoring System Based on a Time Series Database (Post-print)

Authors: Chai Yagang

Date: 2023-10-08T00:00:00+00:00

Abstract

Communication between application systems in large-scale data centers frequently spans multiple machine rooms or multiple core network regions, resulting in substantial increases in network communication quality fluctuations. A corresponding real-time monitoring technical architecture is employed to establish a foundation platform for whole-network network fluctuation monitoring, thereby realizing network fluctuation monitoring of availability, reliability, and user experience.

Full Text

ChinaXiv Collaborative Journal

A Distributed Network Fluctuation Monitoring System Based on Time-Series Databases

Abstract: In large-scale data centers, communication between application systems frequently spans multiple machine rooms or core network zones, resulting in substantial increases in network communication quality fluctuations. This paper introduces a real-time monitoring technical architecture that establishes a foundational platform for network fluctuation monitoring across the entire network, enabling surveillance of availability, reliability, and user experience.

Keywords: time-series database; data center; information network; data monitoring
Classification Number: TP393
Document Code: A
Article ID: 1671-0134(2018)03-036-02
DOI: 10.19483/j.cnki.11-4653/n.2018.03.014
Author: Chai Yagang

1. Business Scenarios and Requirements

As data center system scale continues to expand, inter-application communication frequently crosses multiple machine rooms or core network zones, substantially increasing the probability of network communication quality fluctuations that directly impact normal system operation. Consequently, network-wide fluctuation monitoring has become an indispensable component of foundational platform monitoring. Network fluctuation monitoring primarily focuses on availability, reliability, and user experience through several key aspects: (1) selecting random nodes from each network zone as collection points for continuous monitoring of ICMP probe response round-trip times across machine rooms and network zones; (2) selecting user-side network nodes as collection points for continuous monitoring of response times, statuses, and content for all service domains and critical URLs; (3) enabling real-time queries of historical data and aggregation calculations of monitoring metrics over specified periods, with network responses visualized through various chart formats; and (4) supporting alert policy definitions that trigger notifications via Web Hook, Email, and other channels when collected metrics meet specified conditions. Since network fluctuation monitoring data consists primarily of time-series data, we propose implementing the solution using a time-series database in conjunction with distributed collection tools, messaging systems, and monitoring frontend systems.

2. Current Development of Time-Series Databases

According to November rankings from db-engines.com[1], time-series databases have gained significant prominence. Unlike conventional data, each time-series data point carries a timestamp reflecting measurements at a specific moment. A time-series database (TSDB) is optimized for timestamps or time-series data, specifically designed for tracking, monitoring, aggregating, and processing metrics or events that change over time. These metrics may include server indicators, application performance monitoring data, network performance data, sensor data, events, clicks, market transactions, and various other data types.

Time-series databases are not new innovations; their early applications primarily focused on trading systems for monitoring stock transaction volatility. However, over the past decade, as PC servers have gradually replaced mainframes and minicomputers, and with the rapid development of Internet, IoT, and big data technologies, massive amounts of time-series data, indicators, and events are being generated continuously and everywhere. New business requirements arising from these evolving data sources—including data lifecycle management, real-time querying and aggregation across long time spans, and predictive analysis based on historical data—necessitate corresponding evolution in underlying data infrastructure, requiring distributed time-series databases better suited for Internet-scale applications. In the time-series database landscape, InfluxDB, RRDtool, Graphite, OpenTSDB, Druid, and Prometheus rank among the most prominent solutions.

3. Key Concepts in Time-Series Databases

Using InfluxDB, the most widely adopted time-series database, as an example, several key concepts are essential for understanding its architecture and operation.

First, **field key/field value/field set** represent metric data, where the field key is the measurement indicator field and the field value is its corresponding value; together they form a field set. Field keys are not indexed, so filter queries based on fields require full table scans. Field values can only be strings, floating-point numbers, integers, or boolean types, with each metric data point bound to a timestamp.

Second, **tag key/tag value/tag set** constitute optional index labels, where the tag key is the index label field and the tag value is its corresponding value; together they form a tag set that can be used in WHERE clauses of queries.

Third, **measurement** is analogous to a table in relational databases, storing tags, fields, and corresponding timestamps.

Fourth, **retention policies** define data storage strategies, with permanent storage as the default. Expiration times can be set for data tables, and InfluxDB periodically purges expired data.

Fifth, **database** is a logical concept similar to relational databases, encompassing user permissions, storage policies, and time-series data.

Sixth, **series** represents data sequences, where identical measurements, retention policies, and tag sets constitute a single series. When tables contain tag labels, various permutations and combinations of different tag labels typically form multiple data series, making this the most critical concept in time-series databases.

For network fluctuation metric monitoring, field data primarily consists of ICMP and HTTP request response time data, while collector names, monitoring targets, and return statuses serve as tag data.

4. Real-Time Monitoring Technical Architecture

Given the extensive network collection scope, numerous nodes, short collection intervals, and high data insertion concurrency, a distributed architecture is essential for network fluctuation metric data collection and processing. Collection tools deployed across network-wide collection points upload data in JSON standard format to a messaging system. Reception and processing programs consume messages from the messaging system, process them, and insert the results into the time-series database. The frontend monitoring system retrieves data from the time-series database for display and generates alerts based on customized policies [Figure 1: see original paper].

The architecture leverages HeartBeat as the collection tool, a lightweight data

collector from the Elastic Stack Beats suite. Developed in Go for high concurrency performance, HeartBeat supports three types of heartbeat monitoring (ICMP, HTTP, and TCP), enables dynamic addition and deletion of targets, and supports direct output to Elasticsearch and Logstash as well as output to Kafka and Redis message queues. The messaging system employs Kafka, an Apache Foundation project defined as a distributed stream processing platform commonly used for real-time streaming data pipelines and stream processing applications. Developed in Java with high concurrency support, Kafka accommodates both publish/subscribe messaging scenarios and log storage scenarios.

The time-series database selected is InfluxDB, an open-source project from InfluxData that consistently ranks at the top of time-series database rankings on db-engines.com. The monitoring frontend utilizes Grafana, which supports various data sources including InfluxDB, Elasticsearch, Graphite, and Prometheus, and offers diverse display modes such as charts, tables, and dashboards. Grafana enables alerting through custom alert thresholds and messages [Figure 2: see original paper].

5. Time-Series Data Storage and Query

Network fluctuation monitoring focuses primarily on ICMP and HTTP response result metrics, which are stored in separate time-series databases. Within each database, tables are appropriately partitioned based on data source, time, and other fields, while error responses are replicated into dedicated error tables to facilitate querying.

InfluxDB's query syntax resembles SQL but includes enhancements and optimizations for calculating maximum, minimum, and average values of metrics over specified time ranges. For example, querying ICMP response sequences between network zones involves selecting the average ICMP response time from a specific regional node, with time intervals serving as the aggregation basis. The query statement is as follows:

```
SELECT mean(rtt) FROM icmp_{metrics} WHERE time > now() - 1h GROUP BY time(1m), region
```

Compared to cross-network-zone ICMP response monitoring, HTTP response data for critical website URL monitoring involves more complex metrics, including TCP connection time, DNS resolution time, HTTP response time, and HTTP response status. The meaning of each column is as follows: time represents the timestamp, response_{status} indicates the response status, up indicates whether a response was received, resolve_{rtt} represents DNS resolution time, tcp_{connect} represents TCP connection time, and http_{rtt} represents HTTP response time. The query statement is as follows:

```
SELECT mean(resolve_{rtt}), mean(tcp_{connect}), mean(http_{rtt}) FROM http_{metrics} WHERE
```

6. Summary and Outlook

This paper addresses the practical requirements for monitoring network quality fluctuations across machine rooms and network zones, designing a distributed network fluctuation monitoring system based on time-series databases. By employing a message queue system for metric data pipeline transmission, the architecture decouples collection nodes and data reception processing modules distributed across different machine rooms and network zones, effectively expanding monitoring scope and capacity. Utilizing a time-series database to store network quality metric data and frontend display components for data visualization effectively satisfies the requirements for real-time monitoring of network quality and fluctuations and historical data query visualization.

Future work will focus on further optimizing the system for larger-scale deployments and exploring advanced analytics capabilities for predictive network performance management.

References

[1] Zhi Lin. Time-Series Database Mining Based on Information Theory Networks[J]. Computer Engineering and Applications, 2003(01).
[2] He Huang. Research on Fast Similarity Search Algorithms in Time-Series Databases[J]. Pattern Recognition and Artificial Intelligence, 2003(02).
[3] Siwen Guo. Network Time-Series Data Mining Based on Wavelet Technology[J]. Computer Engineering, 2007(02).

(Author Affiliation: Liaoning Branch, Xinhua News Agency)

Note: Figure translations are in progress. See original paper for figures.