Poster: Towards a Publicly Available Framework to Process Traceroutes with MetaTrace

Matthieu Gouel, Omar Darwich, Maxime Mouchet, Kevin Vermeulen

Poster: Towards a Publicly Available Framework to Process Traceroutes with MetaTrace

Matthieu Gouel
Sorbonne Université
Paris, France

Omar Darwich
LAAS-CNRS, Université de Toulouse, CNRS
Toulouse, France

Maxime Mouchet
Unaffiliated
Paris, France

Kevin Vermeulen
LAAS-CNRS, Université de Toulouse, CNRS
Toulouse, France

ABSTRACT

The objective of this research is to contribute towards the development of an open-source framework for processing large-scale traceroute datasets. By providing such a framework, we aim to benefit the community by saving time in everyday traceroute analysis and enabling the design of new scalable reactive measurements [1], where prior traceroute measurements are leveraged to make informed decisions for future ones [8, 12].

It is important to clarify that our goal is not to surpass proprietary solutions like BigQuery, which are utilized by CDNs for processing billions of traceroutes [6, 10]. These proprietary solutions are not freely accessible to the public, whereas our focus is on creating an open and freely available framework for the wider community.

Our contributions include (1) sharing the ideas and thinking process behind building MetaTrace, which efficiently utilizes ClickHouse features for traceroute processing; and (2) providing an open-source implementation of MetaTrace[1].

We evaluated MetaTrace using two types of queries: predicate queries for filtering traceroutes based on conditions, and aggregate queries for computing metrics on traceroutes. Our results show that MetaTrace is significantly faster compared to alternative solutions. For predicate queries, it outperforms a multiprocessed Rust solution by a factor of 552 and is 3.4 times faster than ClickHouse without MetaTrace optimizations. For aggregate queries, MetaTrace processes 202 million traceroutes in 11 seconds, with its performance scaling linearly with traceroute volume. Notably, on a single server, MetaTrace can perform a predicate query on a 6-year dataset of 6 billion traceroutes in just 240 seconds.

Furthermore, MetaTrace is resource-efficient, making it accessible for research groups with limited resources to conduct Internet-scale traceroute studies.

ACM Reference Format:
Matthieu Gouel, Omar Darwich, Maxime Mouchet, and Kevin Vermeulen. 2023. Poster: Towards a Publicly Available Framework to Process Traceroutes with MetaTrace. In *Proceedings of the 2023 ACM Internet Measurement Conference (IMC '23), October 24–26, 2023, Montreal, QC, Canada.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3618257.3625001

[1]https://github.com/dioptra-io/metatrace

IMC '23, October 24–26, 2023, Montreal, QC, Canada

ACM ISBN 979-8-4007-0382-9/23/10.
https://doi.org/10.1145/3618257.3625001

1 METHODOLOGY

Which processing do we want to perform? The initial consideration before selecting a database was determining the common computations we performed on traceroutes. Our use-cases align with previous studies [5, 7, 12] that focus on computing aggregated statistics, such as the RTT between a source and a destination, or converting IP-level paths from traceroutes into AS-level paths and analyze specific ASes like Tier 1s or CDNs.

The common use cases can be translated into two general query types: aggregate queries and predicate queries. On one hand, an aggregate query calculates summarized information about the data, such as the minimum RTT among a set of traceroutes. On the other hand, a predicate query returns only the traceroutes that meet specific conditions, such as whether the traceroute traverses a particular AS.

Which database to choose? Optimizing the choice of a database for performant aggregate and predicate queries is not overly restrictive. However, we observed that there is no compelling reason to modify or delete traceroute data. Consequently, we sought a database that prioritizes efficient read operations. We found the ClickHouse database [3] to be a promising candidate as it is promoted as a database optimized for reads and aggregations. Moreover, ClickHouse offers a comprehensive set of convenient features for expressing complex aggregate queries. Notably, ClickHouse introduces a powerful feature called "groupArray", which facilitates manipulation of data in aggregate queries.

While the benchmark conducted by ClickHouse itself [4] was compelling, we independently verified its superior performance over standard solutions like MySQL and PostgreSQL, even on simple queries. This confirmed that ClickHouse significantly outperformed these solutions by orders of magnitude in our specific setup. However, it is important to clarify that we are not asserting ClickHouse as the ultimate solution for processing traceroutes. Nevertheless, based on our experience, ClickHouse has proven to be highly reliable and has fulfilled our requirements for several years [8, 12, 13]. We believe it is valuable to share this information with the community. Furthermore, we emphasize that ClickHouse employs a SQL-like language that is easy to learn, requiring minimal effort for the community to adopt such a database. Finally, we have developed an open-source tool[2] that enables the conversion of traceroute data

[2]https://github.com/dioptra-io/pantrace

from one commonly used format [2, 11] to another, including the MetaTrace format, in order to facilitate the integration of multiple traceroute datasets.

Which schema to represent a traceroute? Now that we have selected our database, we need to determine an appropriate schema for storing the traceroutes. Given that ClickHouse is a column-oriented database, we structure our schema based on the fields present in an ICMP reply to a traceroute probe (e.g., TTL, RTT). Additionally, we include fields that uniquely identify the traceroute that generated the probe, such as the flow identifier and traceroute timestamp. In this design, a traceroute is represented as a collection of rows, where each row contains these fields. The flow identifier within each row enables differentiation between load-balanced paths, as encountered in per-flow load balancing scenarios [12].

This data format offers several advantages. First, it is compatible with all existing traceroute tools, serving as the lowest common denominator for platforms and tools that utilize traceroute techniques. These tools typically send TTL-limited probes to generate ICMP TTL exceeded messages from routers along the path. Second, the format is memory-efficient for predicate queries. By loading only the relevant column(s) for a predicate, it becomes feasible to identify the rows that match the predicate efficiently.

How to improve the database performance? Finally, we can leverage ClickHouse's features to minimize the query time and the resource usage. To achieve this, we judiciously select how to sort traceroute data enabling queries to be processed in blocks of rows. ClickHouse facilitates this through its "sorting key" mechanism. It enables efficient utilization of RAM while also optimizing disk usage, as the sorting key also influences the compression quality of the columns.

However, it is essential to note that a table can have only one sorting key, limiting the ability to optimize for predicate queries on arbitrary columns. Assuming we use the sorting key (source, destination, timestamp), our objective is to have a mechanism to sort the data by (predicate, source, destination, timestamp), where the predicate can be a subset of columns or even the result of an aggregate query per (source, destination, timestamp). ClickHouse provides a solution for this through the materialized view feature. It enables us to generate a table from the computed result, thus creating a materialized view for each predicate, significantly enhancing predicate query performance. Finally, since the materialized view is properly sorted, the increase in disk usage remains reasonable.

How to add metadata on traceroutes? Metadata refers to additional information associated with IP addresses, such as ASes or geolocation. ClickHouse offers a feature called "dictionary", which enables the creation of a table containing mappings from IP addresses to corresponding values. Specifically, this dictionary can be implemented as a radix tree that maps IP prefixes to metadata. To ensure the most relevant metadata is used for each traceroute, the radix tree can be parameterized by a date.

2 EVALUATION

On a dataset of 202 million traceroutes from one day of RIPE Atlas[11], MetaTrace takes 25% less disk space than the compressed JSON source data. MetaTrace is able to serve predicate queries 552x faster than a multiprocessed Rust solution, and is 3.4x faster than the ClickHouse database without MetaTrace's optimizations. On aggregate queries, MetaTrace is 62x faster than our Rust solution. Overall, MetaTrace is able to serve both types of queries with a very reasonable 1.6 GB maximum usage of memory. MetaTrace's performance scales linearly with the number of traceroutes. The technical details of the evaluation of MetaTrace can be found in a distinct technical report [9].

3 CONCLUSION

This work highlights the advantages of utilizing MetaTrace, an intelligent utilization of ClickHouse, for traceroute processing. MetaTrace outperforms multiprocessed Python and Rust solutions, as well as the raw ClickHouse database, in terms of query response time for both predicate and aggregate queries. Furthermore, this performance improvement is achieved while utilizing reasonable amounts of memory, CPU, and disk resources. As a result, MetaTrace enables the processing of large traceroute datasets, potentially in combination, within a local environment. This work will hopefully enable us and the community to build reactive traceroute measurement frameworks at Internet scale.

ACKNOWLEDGEMENTS

A research grant from the French Ministry for Armed Forces has made this work possible. We thank Robert Beverly for their careful reading and feedback on earlier versions of this work. Matthieu Gouel is associated with Sorbonne University, CNRS, Laboratoire d'informatique de Paris 6, LIP6, F-75005 Paris, France. Matthieu Gouel has partially carried out the work presented in this extended abstract at LINCS (`www.lincs.fr`). Omar Darwich and Kevin Vermeulen are associated with the LAAS-CNRS, Paul Sabatier University, CNRS, Toulouse, France.

REFERENCES

[1] Mark Allman and Vern Paxson. A reactive measurement framework. In *Proc. PAM*. Springer, 2008.
[2] CAIDA. Archipelago. https://www.caida.org/projects/ark.
[3] ClickHouse. ClickHouse Database. https://clickhouse.yandex, .
[4] ClickHouse. ClickHouse Benchmark. https://benchmark.clickhouse.com/, .
[5] Í. Cunha, P. Marchetta, M. Calder, Y-C. Chiu, B. Schlinker, B. V. A. Machado, A. Pescapé, V. Giotsas, H. V. Madhyastha, and E. Katz-Bassett. Sibyl: A Practical Internet Route Oracle. In *Proc. USENIX NSDI*, 2016.
[6] Ashley Flavel, Pradeepkumar Mani, David Maltz, Nick Holt, Jie Liu, Yingying Chen, and Oleg Surmachev. FastRoute: A Scalable Load-Aware Anycast Routing Architecture for Modern CDNs. In *Proc. USENIX NSDI*, 2015.
[7] Romain Fontugne, Cristel Pelsser, Emile Aben, and Randy Bush. Pinpointing delay and forwarding anomalies using large-scale traceroute measurements. In *Proc. ACM IMC*, 2017.
[8] Matthieu Gouel, Kevin Vermeulen, Maxime Mouchet, Justin P Rohrer, Olivier Fourmaux, and Timur Friedman. Zeph & Iris map the internet: A resilient reinforcement learning approach to distributed ip route tracing. *Proc. ACM SIGCOMM Computer Communication Review*, 52(1):2–9, 2022.
[9] Matthieu Gouel, Maxime Mouchet, Omar Darwich, and Kevin Vermeulen. Towards a publicly available framework to process traceroutes with MetaTrace. September 2023. URL https://hal.science/hal-04198068.
[10] M-Lab. M-Lab. https://www.measurementlab.net.
[11] RIPE NCC Staff. RIPE Atlas: A global internet measurement network. *Internet Protocol Journal*, 18(3):2–26, 2015.
[12] Kevin Vermeulen, Justin P Rohrer, Robert Beverly, Olivier Fourmaux, and Timur Friedman. Diamond-Miner: Comprehensive discovery of the internet's topology diamonds. In *Proc. USENIX NSDI*, 2020.
[13] Kevin Vermeulen, Ege Gurmericliler, Ítalo Cunha, Dave Choffnes, and Ethan Katz-Bassett. Internet scale reverse traceroute. In *Proc. ACM IMC*, 2022.