universität wien

Bachelor Thesis Exposée

Faculty of Computer Science
University of Vienna

A Lightweight Persistence System for Network Traffic Measurements

Name:	Michael Mente
Student ID:	01634435
Student-E-Mail	a01634435@unet.univie.ac.at
Supervisor	Oliver Michel, BSc MSc PhD

Motivation

As never before, the internet plays a central role in our society, driven on the one hand by technological advances and on the other by social conditions such as the COVID19 pandemic. All of these things have led to services supported by cloud computing providers being more heavily utilized than ever before, be it by the increased use of video conferencing tools or by online streaming providers. As a result network traffic is not only increasing in volume, but also in velocity. However, due to the growing amount of network traffic, it is becoming more and more challenging to analyze this data. Currently, traffic is mostly processed live by means of special applications to detect possible issues.
Despite these sophisticated algorithms, it is still possible that a security or performance issue cannot be detected in real time. Nevertheless, there are only limited possibilities to analyze a disturbance after it has already occured [1].
One possibility to analyze a fault at a later point in time is a database system that stores all network traffic. However, this approach is very storage intensive and existing solutions do not scale to meet these high traffic demands. In addition, these systems use, among other things, mostly complex indexes that require a lot of computational effort due to which the ingestion rate is comparatively low.
There are two existing approaches to store the data:
On the one hand, classic SQL databases have the advantage of straighforward data analysis due to their natural structure. Already existing SQL databases optimized for time-series-data have a comparatively high computational overhead and are therefore not fast enough to meet the requirements. One of the fastest relational databases for this type of data is Timescale DB. However, it has been shown that the maximum ingestion rate of about 400K packets per second is not sufficient to log the traffic of a top of rack switch [1].
On the other hand, NoSQL databases offer a potentially higher ingest rate which is one of the most important factors for a project of this kind. Nevertheless, even NoSQL systems are not nearly performant enough to meet the requirements.

My goal is to develop (a prototype of) a specialized storage system that is optimized for high ingestion rates and the reduction of storage requirements in this particular use case. There should also be a simple way to query the data, in a similar way to existing SQL implementations.

A possible use case for this application would be a retrospective analysis of network traffic. For example, a customer of a cloud provider could contact the provider shortly after an incident and ask what exactly caused the disruption. This could potentially increase the robustness of the services provided, as the customer would gain new insights into the network traffic.

Main Challenges

In this thesis, I plan to focus particularly on three main challenges:
One challenge is to store the data fast enough. Ideally, the peak write access rate would be the limit of the hardware. For modern NVME SSDs, the limit for write accesses is about 1GB/s.

The second major issue relates to the efficient storage of network packets. After all, some parts of the packets are similar or even the same. Simple compression methods must be used to reduce the amount of data. One possible approach would be delta-compression, which is well suited for the timestamps of the packets, for example. In addition, aggregation of packets based on IP 5-tuple offers another way to use storage space efficiently. For this intended use, this is common practice and has already been done efficiently in hardware in earlier research [2].

The third challenge is to query the data selectively. At this point it is necessary to solve simple queries to narrow down the searched packets. It is important to store the data in a way for it to be restored to

the original traffic.

Lastly, it is necessary to find a balance so that one does not encounter a bottleneck. Either the CPU can be the limiting factor due to too complex and computationally intensive compression procedures, or the write rate of the storage devices is not high enough to store such amounts of data.

Methods

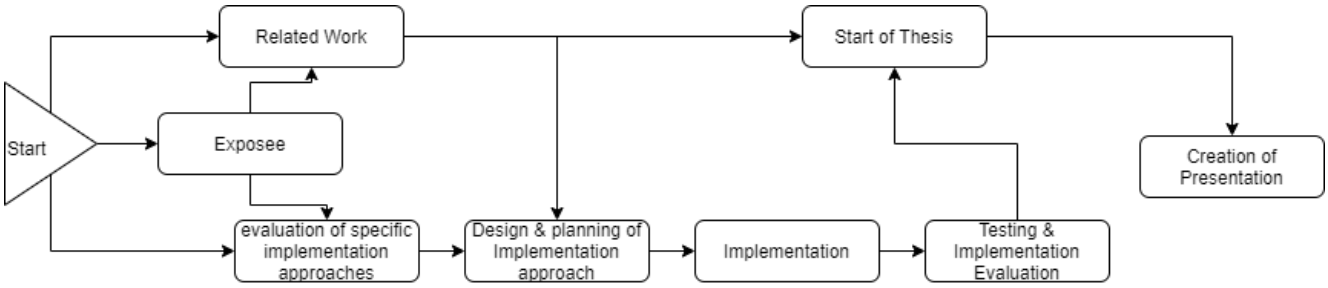
To achieve these goals, I plan to use the following methods:
I will implement a prototype as a proof-of-concept which subsequent functionality will be demonstrated through experiments. This prototype should meet the following requirements:
The ingestion rate should be higher than already existing general-purpose database solutions which I plan to achieve by domain-specific optimizations. Despite compression, it should be possible to query the data in a meaningful way.
To perform these experiments, I will use a dataset of network packages from Equinix - a data center in the USA – which were supplied by the CAIDA organization. By using this sample of real traffic, I aim to achieve more realistic test results.

Regarding the compression algorithms I will on the one hand use common methods of timeseries databases - for example delta compression and run length encoding [3] - on the other hand there is potential for optimization as only network packets are stored. Here it is possible to use properties of the network topology to optimize certain fields (e.g. the IP address) by using dictionary encoding.
As for the retrieval methods of the data, I would like to support simple queries. For example, it should be possible to request information about the data, e.g. the packets per second for a certain period of time. Also packets can be queried directly for example "all packets with a certain IP address and protocol". Furthermore it should be possible to restore the intercepted traffic (or a certain part of it) as a PCAP file.

The first two challenges (ingestion rate, compression rate) can be measured and shown with quantitative methods. Furthermore, I aim to draw comparisons to already existing literature.
The third challenge is testing the queries, which can be tested with qualitative analysis, by testing it against classic database implementations and by considering the purpose of application.

Time Schedule

	Activity	**From**	**To**	**Duration**
1	Expose	01.03.2021	18.03.2021	17
2	Related Work	13.03.2021	27.03.2021	14
3	Evaluation of specific implementation approaches	13.03.2021	20.03.2021	7
4	Design & planning of Implementation approach	20.03.2021	30.03.2021	10
5	Implementation	01.04.2021	13.05.2021	42
6	Testing & Implementation Evaluation	13.05.2021	27.05.2021	14
7	Start of Thesis	13.05.2021	17.06.2021	35
8	Creation of Presentation	17.06.2021	21.06.2021	4



Preliminary Outline

1. Introduction
2. Motivation & Related Work
   2.1. Network Monitoring & Analytics
   2.2. Record Persistence & Retrospective Queries
3. Implementation
   3.1. Architecture
   3.2. Data Pipeline
   3.3. Record Compression
   3.4. Query Module
4. Evaluation
   4.1. Record Ingestion
   4.2. Storage Requirements
   4.3. Query Capabilities
5. Conclusio

Bibliography

[1] Michel, O., Sonchack, J., Keller, E. and Smith, J.M., 2019, March. PIQ: Persistent interactive queries for network security analytics. In *Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization* (pp. 17-22).

[2] Sonchack, J., Michel, O., Aviv, A.J., Keller, E. and Smith, J.M., 2018. Scaling hardware accelerated network monitoring to concurrent and dynamic queries with* flow. In *2018 {USENIX} Annual Technical Conference ({USENIX}{ATC} 18)* (pp. 823-835).

[3] Lockerman, J. (2021, March 10). Time-series compression algorithms, explained. Retrieved from https://blog.timescale.com/blog/time-series-compression-algorithms-explained/