Network Security Analytics

Sunil Pedapudi Matthias Vallentin
{sunil, vallentin}@cs.berkeley.edu

1 Motivation

Incident response, post-facto forensics, and network troubleshooting rely on the ability to quickly extract relevant information. To this end, security analysts and network operators need a system that *(i)* allows for directly expressing a query using domain-specific constructs, and *(ii)* delivers the performance required for interactive analysis. The database community offers an abundance of tuned niche solutions that claim to address the latter aspect, but the overwhelming majority of these works are not available as stable open-source packages, fail to deliver the advertised performance with workloads from our domain, or have inherent limitations in the expressiveness of their data model.

Therefore, we started to build our own system that supports the field's common idioms and exhibits a scalable architecture capable of handling the large, continuously arriving data. In previous work, we designed and implemented a prototype whose storage layer was not well-suited for distributed deployment. In this project, we aim to continue this effort by redesigning and implementing the building blocks of the underlying storage system while considering cutting-edge distributed database technology.

Network security analysis generates large volumes of rich-typed, semi-structured log data [4]. Existing tools do the heavy lifting of extracting this fine-grained data [5] which we would like to archive permanently. In the case of incident response, the analysis procedure often begins with a piece of intelligence, such as "this MD5 hash of a file is malware," or "connections to IP address X contain botnet traffic." An analyst uses this information to identify affected machines in the own network, quarantine them, and investigate the collateral damage. This process involves filtering the data (project, select) and computing aggregates (sum, quantile, top-k, unique) interactively. The analysis proceeds in an ad-hoc fashion and may involve several refinements until the full extent of an incident has been determined. Our goal in this project is to build a system that can handle this use case with real-world data from UC Berkeley's 10-Gbps upstream link.

2 Roadmap

2.1 Storage

The storage component characterizes expressiveness and performance of the analytic system. We aim to explore a non-relational design that allows for rich data models and lends itself to distributed query processing. This element of the project identifies what basic blocks of storage are necessary to preserve application-specific semantics as network events are committed to persistence.

Then, given an abstract model of data, we aim to identify an architecture of storage that meets interactive query response times with simultaneous high-throughput data archival. The goal is to create a hybrid system that supports queries on historical data as well as incoming data, and to do this, the naturally parallel nature of non-relational data stores provide an ideal starting point. We aim to expand on these systems by providing aforementioned indices and efficient range scans. This architecture should provide modularity and horizontal scalability such that different workloads can adopt varying deployment sizes without needing to modify the core design of our system.

2.2 Basic Query Processing

As a short-term goal, we intend to support selections and projections over data with interactive query response times. This suggests an exploration of distributed query processing. We will present a SQL-like query interface to the end user and translate it to a distributed query execution plan.

3 Evaluation: Forensics and Incident Response

To validate the effectiveness of our system, we will present a data exploration task and show how we facilitate navigation of the data in a flexible manner. This suggests we identify a metric of interactivity for network forensics as well as describe enabling primitives provided for pinpointing network events. If mature, we will show incident response schemes that can programatically identify events of forensic interest.

4 Related Work

DBMS and DSMS Traditionally, data base management systems support historical queries over already-archived data while data stream management systems support live queries that incorporate newly arriving data after a query is issued. Our work merges these into a hybrid system [6] much like TelegraphCQ [2].

OLAP An alternative perspective of the query workload that our system supports is described by online analytical processing systems which, given a database warehouse of bulk-loaded data, allow for immutable data processing. However, our aim differs fundamentally from existing OLAP systems in supporting side-by-side execution of both queries and data archival.

NoSQL To meet performance goals of online data reporting, we will explore works of “Not only SQL” or “non-relational” storage engines [1] for inspiration towards horizontal scalability. Ideally, NoSQL storage engines allow for simultaneous reads and writes of parallel data streams due to the data access patterns they allow. For the same reason, we surmise that NoSQL storage will also help address the challenge of performing data archiving in light of expansive amounts of incoming data.

Network Security Finally, we explore Time Machine [3] which aims to record network traffic for future inspection. Time Machine is similar in goal to our project in that it provides high-performance network traffic capture, indexing, and retrieval; we extend these concepts to arbitrary events that can stem from a wide range of sources. Further, a major limitation of Time Machine is that it works only on a local machine while we aim to scale in a cluster setting.

References

[1] R. Cattell. Scalable SQL and NoSQL Data Stores. *SIGMOD Record*, 39(4):13, 2010.

[2] S. Chandrasekaran, O. Cooper, A. Deshpande, M. J. Franklin, J. M. Hellerstein, W. Hong, S. Krishnamurthy, S. Madden, V. Raman, F. Reiss, and M. Shah. TelegraphCQ: Continuous Dataflow Processing for an Uncertain World. In *Proceedings of the 1st Biennial Conference on Innovative Data Systems Research (CIDR'03)*, Asilomar, CA, January 2003.

[3] G. Maier, R. Sommer, H. Dreger, A. Feldmann, V. Paxson, and F. Schneider. Enriching network security analysis with time travel. In *SIGCOMM '08: Proceedings of the 2008 conference on Applications, technologies, architectures, and protocols for computer communications*, New York, NY, USA, August 2008. ACM Press.

[4] S. Melnik, A. Gubarev, J. J. Long, G. Romer, S. Shivakumar, M. Tolton, and T. Vassilakis. Dremel: Interactive Analysis of Web-Scale Datasets. *Proceedings of the VLDB Endowment*, 3(1-2):330–339, September 2010.

[5] V. Paxson. Bro: A System for Detecting Network Intruders in Real-Time. *Computer Networks*, 31(23–24):2435–2463, 1999.

[6] F. Reiss, K. Stockinger, K. Wu, A. Shoshani, and J. M. Hellerstein. Enabling real-time querying of live and historical stream data. In *SSDBM '07: Proceedings of the 19th International Conference on Scientific and Statistical Database Management*, page 28, Washington, DC, USA, 2007. IEEE Computer Society.