Gigascope: A Stream Database for Network Applications

Authors: Cranor, Johnson, Spataschek (AT&T Labs), Shkapenyuk (CMU)
Presented by: Brian Agala
Overview

• Problem
• Goals
• Background: Data Streams
• Gigascope Data Stream Management System
• Conclusions
Problem: Managing a Large Data Communications Network

• Requires constant network monitoring
• Decentralized → Difficult to manage
• Analyze network trace dumps
• Limited set of network monitoring reports
Goals

Develop a network data analysis tool which has:

- Speed and flexibility that network analysts require
- Provides structured querying environment to make complex analysis easy to control
Goals

Create a data analysis engine that will be used in many settings:

• traffic analysis
• performance monitoring
• debugging
• protocol analysis and development
• router configuration
• intrusion detection
• network monitoring
Data Streams: Why Now?

- Haven’t data feeds into databases always existed? Yes
  - Modify underlying databases and data warehouses
  - Complex queries are specified over stored data

- With traditional data feeds
  - Simple queries needed in real-time
  - Complex queries performed offline
Data Streams: Real-Time Queries, High-Volume and High-Velocity Data

- Two recent developments: application and technology driven
  - Need for sophisticated real-time queries/analyses
  - Massive data volumes of transactions and measurements

Massive volumes of data ... arriving at high-velocity ... with the need for real-time queries

Queries
Databases vs Data Streams

Database Systems
• Relation: tuple set
• Data Update: modifications
• Query: transient
• Query Answer: exact
• Query Evaluation: arbitrary

Data Stream Systems
• Relation: tuple sequence
• Data Update: appends
• Query: persistent
• Query Answer: approximate
• Query Evaluation: one pass
Gigascope: Data Stream Management System (DSMS) for Network Applications

• Designed for monitoring high-rate data streams
  • Pure stream database (no stored relations or continuous queries)
  • Pipelined operators that rely on properties of the stream

• Uses SQL-like language, named GSQL
  • Input is a data stream, output is a data stream

• Simplicity of implementation, does not transform input data stream into a windowed table, operate on data stream directly
The **GSQL** Language

- Supports selection, join, aggregation, and stream merge
- GSQL processor is a code generator, translating the query to C or C++ code resulting in a fast execution system
- **Example 1:** Get destination IP, port, and timestamp from TCP packet on the first Ethernet interface card
  
  ```
  DEFINE { query_name tcpDest0; }
  Select destIP, destPort, time From eth0.TCP
  Where IPVersion = 4 and Protocol = 6
  ```

- **Example 2:** Combine streams from multiple sources into a single stream
  
  ```
  DEFINE { query_name tcpDest; }
  Merge tcpDest0.time : tcpDest1.time
  From tcpDest0, tcpDest1
  ```
Gigascope Architecture

- Two layer architecture for early data reduction
  - High level queries for expensive processing (High-level Filtering, Transformation, and Aggregation – HFTA)
  - Fast lightweight data reduction queries (Low-level Filtering, Transformation, and Aggregation – LFTA)
    - Possible to push the query as far down as the NIC as an optimization
Gigascope: Hidden P2P Traffic Detection

• Business Challenge: AT&T IP customer wanted to accurately monitor peer-to-peer (P2P) traffic within their network

• Previous Approach: Using TCP port number found in Netflow data

• Issues: P2P traffic might not use known P2P port numbers

• Solution:
  • Use Gigascope to search for P2P related keywords within each TCP datagram
  • Identified 3 times more P2P traffic than when using Netflow
Gigascope: Web Client Performance Monitoring

- **Business Challenge:** AT&T IP customer wanted to monitor latency observed by clients to find performance problems

- **Previous Approach:** Measure latency from “active clients” that establish network connections with servers

- **Issues:** Use of “active clients” is not very representative

- **Solution:**
  - Use Gigascope to track TCP synchronization and acknowledgement packets
  - Report round trip time statistics: latency

Brian Agala 10/28/2014
Gigascope: Other Applications

Desired goals for Gigascope:

• traffic analysis (E.g. Hidden P2P Traffic Detection)
• performance monitoring (E.g. Web Client Performance Monitoring)
• debugging
• protocol analysis and development
• router configuration
• intrusion detection
• network monitoring
Conclusions

• Querying and finding patterns in massive streams is a real problem with many real-world applications
  • Need for sophisticated real-time queries
  • Massive data volumes of transactions

• Fundamentally rethink data management issues under stringent constraints:
  • Single-pass algorithms with limited memory resources
  • Resource limitations at low-level

• Important to think of end-to-end architecture