Gigascope: High Performance Network Monitoring with and SQL Interface

Chuck Cranor, Yuan Gao, Theodore Johnson, Vlaidslav Shkapenyuk, Oliver Spatscheck
{chuck,ygao,johnsont,vshkap,spatsch}@research.att.com
AT&T Labs – Research

Overview

Operators of large networks and providers of network services need to monitor and analyze the network traffic flowing through their systems. Monitoring requirements range from the long term (e.g., monitoring link utilizations, computing traffic matrices) to the ad-hoc (e.g. detecting network intrusions, debugging performance problems). Many of the applications are complex (e.g., reconstruct TCP/IP sessions), query layer-7 data (find streaming media connections), operate over huge volumes of data (Gigabit and higher speed links), and have real-time reporting requirements (e.g., to raise performance or intrusion alerts).

We have found that existing network monitoring technologies have severe limitations. One option is to use TCPdump to monitor a network port and a user-level application program to process the data. While this approach is very flexible, it is not fast enough to handle gigabit speeds on inexpensive equipment. Another approach is to use network monitoring devices. While these devices are capable of high speed monitoring, they are inflexible as the set of monitoring tasks is pre-defined. Adding new functionality is expensive and has long lead times. A similar approach is to use monitoring tools built into routers, such as SNMP, RMON, or NetFlow. These tools have similar characteristics – fast but inflexible.

A further problem with all of these tools is their lack of a query interface. The data from the monitors are dumped to a file or piped through a file stream without an association to the semantics of the data. The burden of managing and interpreting the data is left to the analyst. Due to the volume and complexity of the data, the burden can be severe. These problems make developing new applications needlessly slow and difficult. Also, many mistakes are made leading to incorrect analyses.

Gigascope

In Gigascope, we take a different approach. We provide an SQL interface to the network monitoring system, greatly simplifying the task of managing and interpreting a stream of data. The clear semantics of the data streams allow us to perform aggressive optimizations, such as executing most or all of a query on the Network Interface Card (NIC).

SIGMOD 2002, June 3-6, 2002, Madison, Wisconsin, USA.

The Gigascope architecture consists of a *stream manager* and a *registry*. Data stream sources are low-level queries (*LFTAs*) which monitor network interfaces, either through libpcap or in the NIC. The LFTAs provide data streams to the stream manager, which routes them to higher-level query nodes (*HFTAs*), or to applications. HFTA nodes also provide data streams to the stream manager. All of the FTAs (both low-level and higher-level) provide the *schema* of their output to the registry, including the attribute names, their data types, the query which the FTA executes, and temporal properties of the attributes (which enables many optimizations).

When a user submits a set of queries, they are analyzed by the system to determine which parts should execute as an LFTA and which as an HFTA. After the queries are split, they are translated into executable code. HFTAs are implemented as separate processes using templatized operators written in C++. LFTAs are translated into C for linking and execution in a run time system (RTS).

To monitor Gigabit networks, we have written a replacement RTS for the Tigon Gigabit Ethernet NIC. When the replacement RTS receives a packet from the network, it presents the packet to a set of processing modules. These modules can perform arbitrary processing (within resource constraints) and produce zero or more output tuples (i.e., their output data stream) for transmission to the host computer. We translate the LFTAs into C modules that follow the API expected by the RTS, link the package into an executable, and load it on the NIC.

We have written a collection of templatized push-based operators in C++ for evaluating the higher-level queries. At the time of writing, these operators are a selection/projection operator, an aggregation operator, a stream merge operator, and a special operator that emulates a network protocol. These operators make use of the temporal properties of the attributes in a stream to optimize processing (e.g., emit aggregates as soon as possible). Given the schema of a stream and the query evaluated on it, we can deduce temporal properties of attributes of the output stream. Thus, both LFTAs and HFTAs produce data streams with temporal properties and we can compose complex queries using stream operators.

Gigascope is a stream database; the input is a stream of packets and the output is a stream of processed tuples. We found that modeling network packets as data streams to be crucial to aggressive optimizations such as executing part or all of a query in the NIC. This optimization is essential for querying a high speed network, as most unnecessary memory copies can be avoided.