OSNT: Traffic Monitoring Architecture

Gianni Antichi

January 20, 2014

The OSNT traffic monitor provides four functions:

- packet capture at full line-rate
- packet filtering permitting selection of traffic-of-interest
- high precision, accurate, packet timestamping
- statistics gathering

Figure 1 illustrates the architecture of the monitoring pipeline that provides the functionality enumerated above. The 5-tuple (protocol, IP address pair and layer four port pair) extraction is performed using an extensible packet parser able to recognize both VLAN and MPLS headers along with IP in IP encapsulation. Further flexibility is enabled by extending the parser implementation-code as required.

A module positioned immediately after the Physical interfaces and before the receive queues timestamps incoming packets as they are received by hardware. Our design is an architecture that implicitly copes with a workload of full line-rate per port of minimum sized packets. However this will often exceed the capacity of the host-processing, storage, etc., or may contain traffic of no practical interest. To this end we implement two traffic-thinning approaches. The first of these is to utilize the 5-tuple filter implemented in the "Core Monitoring" module. Only packets that are matched to a rule are sent to the software, while all other packets are dropped. The second mechanism is to record a fixed-length part of each packet (sometimes called a *snap-length*) along with a hash of the entire original packet. The challenge here is that if a user is interested in all packets on all interfaces it is possible to exhaust the host resources.

As for the software side, we provide a python-based GUI that allows the user to interact with the HW components (*e.g.*, enable cut/hash, set filtering rules, check statistics). A C-based application that comes with it records the received traffic in both PCAP or PCAPNG format. This allows offline use of common libpcap-based tools (*e.g.*, TCPDump, Wireshark.) These tools do not work directly with OSNT: the device driver secures performance by bypassing the Linux TCP/IP stack. We refer the reader to the OSNT website for further information about the software API.

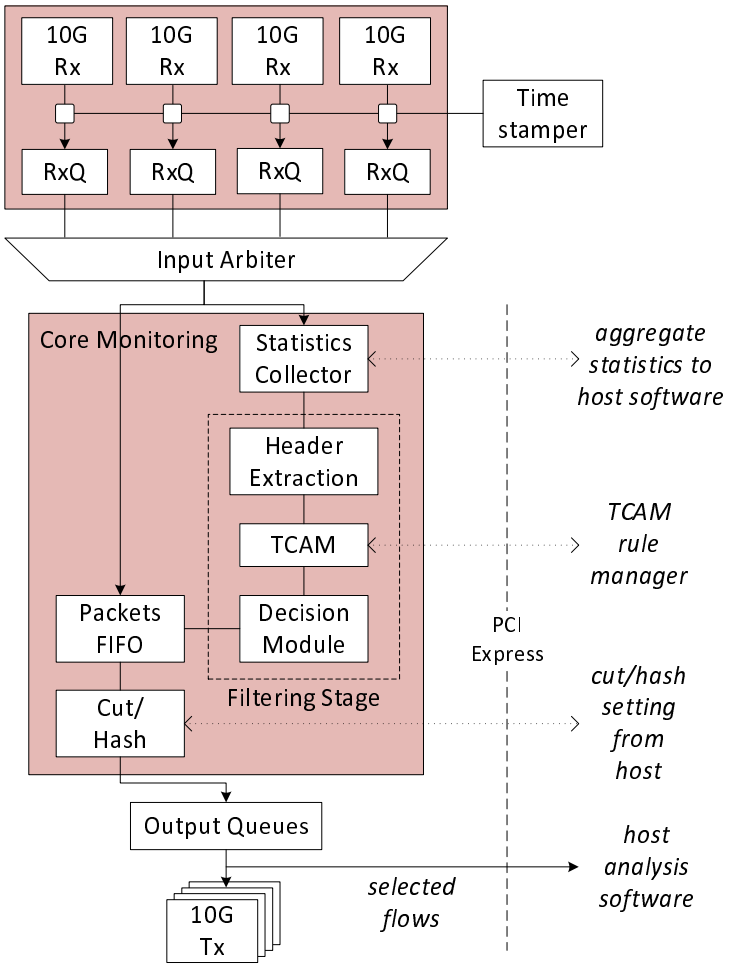

Figure 1: The architecture for OSNT traffic monitoring system.

0.1 Timestamping

Providing an accurate timestamp to (incoming) packets is a critical objective of the traffic monitoring unit. Packets are timestamped as close to the physical Ethernet device as possible so as to minimize FIFO-generated jitter and permit accurate latency measurement. A dedicated timestamping unit stamps packets as they arrive from the physical (MAC) interfaces. Each packet is appended with a 64-bit timestamp.

Motivated by the need to have minimal overhead while also providing sufficient resolution and long-term stability, we have chosen to use a 64-bit timestamp divided into two parts, the upper 32-bits count seconds, while the lower 32-bits provide a fraction of a second with a maximum resolution of approximately 233ps; the practical prototype resolution is 6.25ns. Integral to accurate timekeeping is the need to correct the frequency drift of an oscillator. To this end, we use Direct Digital Synthesis (DDS), a technique by which arbitrary variable-frequencies can be generated using synchronous digital logic. The addition of a stable pulse-per-second (PPS) signal such as that derived from a GPS receiver permits both high long-term accuracy and the synchronization of multiple OSNT elements. The selection of a timestamp with this precision was a conscious effort on our part to ensure the abilities of the OSNT design are at least as good as the currently available commercial offerings.