Monitoring very high speed links

Gianluca Iannaccone, Christophe Diot
Sprint ATL
{gianluca,cdiot}@sprintlabs.com

Ian Graham
University of Waikato
ian@waikato.ac.nz

Nick McKeown
Stanford University
nickm@stanford.edu

Abstract— **Our goal is to build passive monitoring equipment for use at 10Gb/s (e.g. 10GE and OC-192) and above. We already have in place an OC-48 passive monitoring system for capturing and storing a detailed record for every packet. But because of constraints on storage and bus bandwidth this will not be feasible at 10Gb/s and above. Therefore, taking advantage of the fact that packets can be considered as belonging to flows, our system will store per-flow records that are created at the time of capture, and stored alongside small per-packet records. This way storage requirements can be reduced several-fold. Results indicate that it will be possible to capture and store detailed flow information at OC-192 without losing much information compared to our OC-48 packet traces.**

I. Introduction

PASSIVE monitoring involves tapping the link on which data needs to be collected, and recording to disk either complete packets, or just packet headers and timestamps indicating their arrival time.

In this paper we consider how a currently deployed OC-3 and OC-48 monitoring system deployed at Sprint [1] can be scaled to monitor OC-192 links. We will use the OC-48 system as a starting point for our discussion. In short, an optical splitter copies all of the data on the link, which is received by a packet capture card [2] on a PC. Timestamps recorded by the capture card are synchronized to a GPS signal. Packets are temporarily stored on the capture board and then sent to the PC main memory over the PC's PCI bus. We believe this system to be representative of most passive monitoring systems.

Collecting packet traces at higher link speeds is difficult for several reasons:

- PCI bus throughput is already challenged at OC-48 (as explained in [1], the PCI bus is crossed twice for any data transfer: once from the capture board to the main memory, and a second time from the main memory to the hard disk). A new bus technology needs to be used at 10 Gbps.
- We currently collect several terabytes of data per day in a POP at OC-48 speed. At OC-192, the storage capacity must increase by a factor of four, along with the challenge of managing such an enormous data set.
- Memory access speeds do not increase as quickly as the link speed. In fact, memory access speeds have not increased much in the past 5 years and we can not rely on technology improvements in this area for the next generation monitoring tools.
- Disk array speed cannot keep up with link bandwidth. At OC-192 speed, a packet-level trace would require a disk bandwidth of roughly 250 Mbytes/sec (assuming 300 byte packets and a 64 byte long packet records).

The problem is not only limited to individual links. In the future, we hope to see the deployment of monitoring facilities inside the routers. This would be preferable to external monitoring for several reasons. First, if we are interested in how long packets are queued inside a router, it is difficult to use an external monitor. (We would need two - one at ingress and one at egress - with precisely synchronized timestamps). On the other hand, a router can easily measure how long a packet is queued for all interfaces simultaneously. Second, if we are interested in the path taken by packets, we would like to know which port a packet entered and left the router. Again, this is difficult unless we have a precise and uptodate snapshot of the router's forwarding table, or unless we monitor all of the router's ports. Third, it is invasive to splice a link and insert a passive monitor. It would be preferable for monitoring to be integral with the routing equipment.

But if we wish to add monitoring equipment inside the router, we will encounter even more problems of scalability, in terms of data rate, storage, and processing. For example, the aggregate data rate of backplanes are as high as a few hundred gigabits per second today, and will exceed a terabit per second soon. This creates enormous challenges for processing and storage elements. Furthermore, routers are generally built to have the highest capacity for a given volume and power consumption. Monitoring equipment - particularly if it includes significant storage - is bulky and consumes a lot of power. It is difficult to see how it will fit within the router, without compromising its perfor-

mance. Worse still, because newer routers use switched backplanes, rather than shared backplanes, not all packets are seen at any single point of the backplane. As a result, it seems unlikely that routers will include extensive monitoring functions any time soon. And so in this paper, we will focus on the development of a passive monitoring infrastructure that is external to the routers.

Our goal is to define a passive monitoring infrastructure that can be deployed in the near future (OC-192 deployement has already started at most tier 1 ISPs). However this infrastructure is deployed it will have to do some computation on-line so as to minimize the amount of data stored locally. But the computation must be simple – at OC-192 speed (10 Gbps) a new packet arrives every 240ns on average (assuming 300-byte packets). This allows only 360 instructions per packet on the fastest processor (ignoring the – often significant – time that it takes for the data to enter the processor). This reduces to just 90 instructions per packet for OC-768 (40 Gbps) links.

Thus, we define the requirements of a high speed passive monitoring infrastructure:

- The system must perform some information processing at the time data is collected, although it's clear we'll have to accept some severe limitations on what can be computed.
- The system must store the minimum amount of information. Much can be achieved by data compression before data is forwarded to disk servers for archiving. We believe that storing data as IP flows can result in significant compression.
- Sampling might be required in addition to compression [4]. Further investigation is needed in this area to identify appropriate sampling techniques to be used in conjunction with a flow-based trace compression.

In this document, we discuss the compression ratio that can be achieved by storing IP data as flow traces instead of packet traces. And we discuss the feasibility of identifying flows and constructing flow digests in hardware on the capture board, at OC-192 and OC-768 rates. Hardware design elements identified in this paper can either be applied to stand-alone hardware or to hardware embedded in a router.

II. From packet traces to flow traces

A. Flow based compression

In the current monitoring system, we store the first 44 bytes of each packet, that include the IP and transport headers (TCP or UDP), together with a 8 byte timestamp and with HDLC framing information (12 bytes). The fixed size record for each packet is 64 bytes.

In the proposed solution, we will instead maintain a per-flow record, followed by a record for each packet in the flow. We will identify the (application) flow using the classical 5-tuple definition, i.e. source address, destination address, source port, destination port, and protocol type.

Our goal is to store as little data as possible, without compromising the information content of the trace. So we must first decide what information must be kept in the flow trace. We have identified the following information:

- The parameters of the flow that are constant for all packets. These include the 5-tuple and the flow starting time.
- The information that needs to be kept for each packet, i.e. the packet arrival time (offset), packet size, IP id, ToS, time-to-live, sequence numbers (if any) and TCP flags.

There are 2 parts to a flow record. The information about the flow, which is common to all packets in the flow (Figure 1):

- timestamp giving the second when this flow started (32 bits);
- protocol number (8 bits);
- flags (8 bits), in particular we have the Last Record (LR) flag that is used to specify if the current record is the last record related to a given flow;
- record number (16 bits), to enumerate the number of records that constitute a single flow (see section II-C for explanation);
- source IP address (32 bits);
- destination IP address (32 bits);
- source port number (16 bits);
- destination port number (16 bits);
- initial sequence number (32 bits);
- initial acknowledgement number (32 bits);

Note that the timestamp field is used as a base reference for the timestamp of each packet. This is why it has a granularity of one second. Moreover, the shaded areas in Figure 1 represent those field that can be removed depending on the protocol (i.e. port numbers and sequence numbers).

In summary the length of the flow record is 28 bytes for TCP, 20 bytes for UDP and 16 bytes for other protocols.

The second part of the flow trace is made of packet records. For each packet of the flow we add the following information (Figure 1):

- timestamp, a fixed point value with 24 fractional bits that gives the time offset in seconds from the previous packet, or the flow start time (32 bits);
- packet length in bytes (16 bits);
- packet identification (16 bits);
- type of service bits (8 bits);
- time-to-live (8 bits);
- TCP flags such as SYN, FIN, etc. (8 bits);

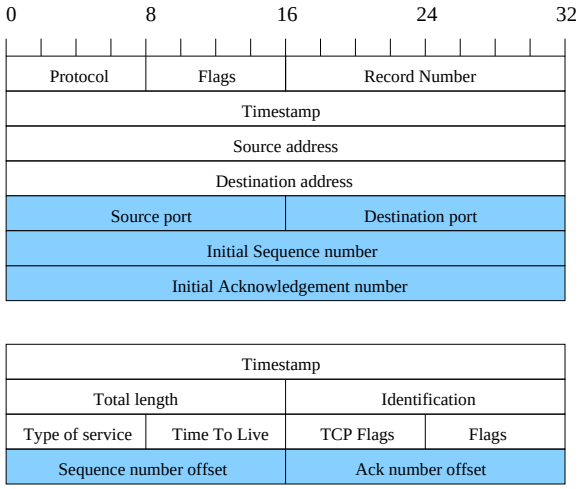

Fig. 1. Flow and packet records.

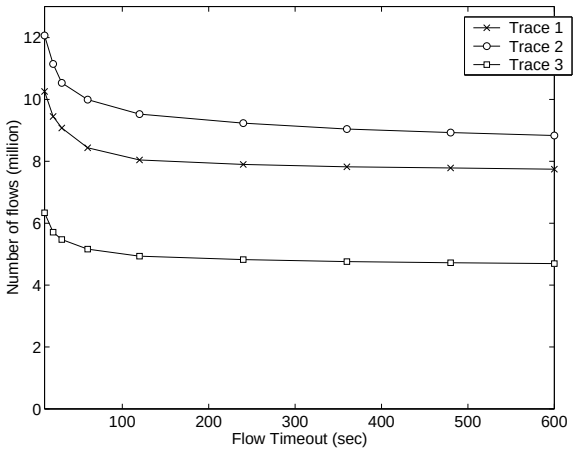

Fig. 2. Evaluation of the flow termination time-out.

- packet record flags, in particular we have the Last Packet (LP) flag that is used to identifiy the last packet belonging to a given flow record (8 bits);
- sequence number offset from the previous packet or the initial sequence number of the flow (16 bits, signed integer);
- acknowledgment sequence number offset from the previous packet or the initial acknowledgement number of the flow (16 bits, signed integer);

In order to investigate the compression ratio achievable with this technique we have compressed some of the OC-3 packet traces collected in [1] using the above flow-based record. Depending on the flow characteristics, a flow trace is between 3 and 4 times shorter than the equivalent packet trace[1]. The compression ratio depends on the number of packets found on an OC-3 link, on the number of flows and on the flow size distribution[2]. Additional measurements carried out on traces from an OC-48 backbone link showed an average compression of similar magnitude for more than 20 GBytes of trace data.

Note that the current state of the art technology allows us to collect packet traces at full rate OC-48 [2]. A compression rate of four would allow us to monitor OC-192 links with the same technology, or for OC-768 links with the next generation bus and disk technology.

B. Flow termination

A major issue is to decide when a flow has terminated. Some flows explicitly indicate that they have completed, for example the FIN or RST segments sent by TCP flows. But some flows give no explicit indication; e.g. UDP flows, or TCP flows for which which no FIN or RST is sent (perhaps because of a poor implmentation, because the trace finished before the flow completed, or because of a network or host failure). And even if there is a end of flow packet, retransmitted packets can be received after the flow appears to have finished. In order to achieve higher compression ratios, data related to a flow should be archived on the host PC only when the flow has terminated.

In our proposed scheme, we have considered using either a static inactivity timeout to determine that a flow has finished [5], or an adaptive timeout based on the characteristics of the connections that are collected [6]. We have decided to follow the approach proposed in [5] with a static timeout value. To determine what this timeout value should be, we have processed our packet traces off-line and counted the number of flows with different time-out values. The result is shown in Figure 2. Five traces were processed. The figure shows the number of flows over a 2 hours period for 3 of these traces. As expected, as the timeout decreases the total number of flows we count in the trace increases. This is due to the fact that with a short timeout we may count the same flow more than once.

From Figure 2 we can see that a timeout value in the range of 60-120 seconds gives a reasonable estimate of the number of flows. However with a timeout value of 60 seconds, the monitoring system must keep on-line state for a number of flows in the range of 60 to 100 thousand.

Therefore, we have a trade-off between the accuracy of the flow termination timeout and the amount of on-line state to maintain in the onboard memory. A short timeout value would reduce the compression ratio causing some flows to be spread over different flow records but at the same time it would also reduce the amount of onboard memory required (e.g. with a 10 seconds timeout the capture board would need to keep state for 10 to 25 thousand

[1] We have run our compression tool on 5 different 24 hour traces and obtained the following compression rates: 3.76, 3.75, 4.02, 3.63, 3.20

[2] Details about these data can be found on the Sprint IP monitoring web site: www.sprintlabs.com/Department/IP-Interworking/Monitor

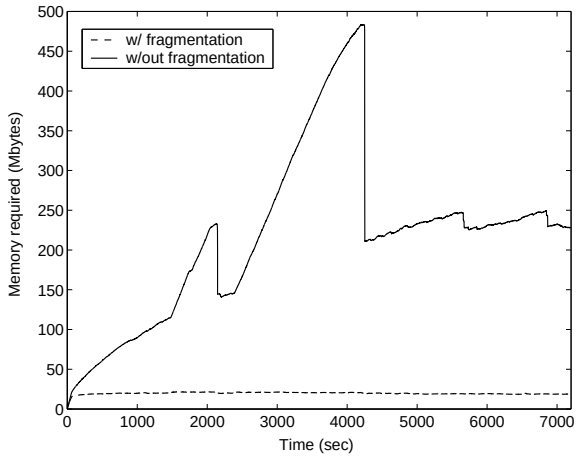

Fig. 3. Evaluation of the memory size required to store flow traces (with the flow termination timeout of 60s).

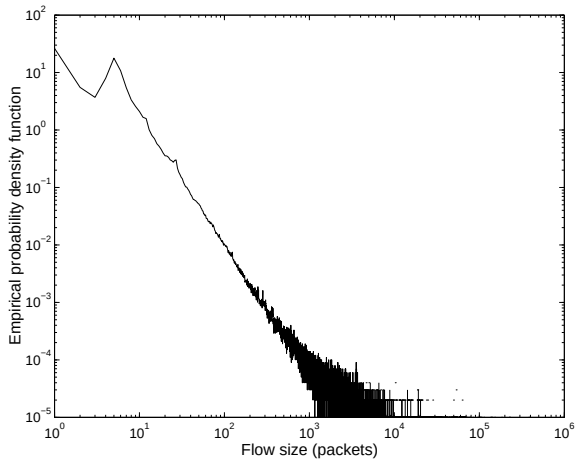

Fig. 4. Flow size distribution in packets (log-log scale).

flows only).

Note that the value of the timeout is not crucial for the trace collection. A short timeout will not cause the system to lose any information. As flow traces are archived, it is possible to post-process the traces to concatenate flows if they have been incorrectly stored in two or more separate flow records.

Further investigation is also needed to account for specific pathological traffic patterns (e.g. SYN attacks, port scans) that can make the number of flows to increase dramatically.

C. Memory requirements

We have designed a program that computes the amount of on-board memory that is necessary to record all flows until completion and to transfer this record once the flow is finished. We used a 60 second timeout to define flow termination. The plain curve on Figure 3 shows the amount of memory for one of the five traces we have processed.

To explain this result, we need to look at the distribution of flow size in packets on the analyzed trace (Figure 4). The peaks can be explained by the presence on the link of very long flows carrying very large numbers of packets. Such flows require a large amount of memory before they terminate and before the recorded information can be moved to main memory. This behavior explains the sudden decrease in memory requirements at time 2000 and 4000 in Figure 3 (the decrease corresponds to the memory freed by the transfer of the large flow record to the PC).

Keeping full flow records on the capture board is consequently a bad idea because (i) it would require a huge amount of fast access memory on the data capture board and (ii) it would make the traffic on the bus between the capture board and the main memory be very bursty, eventually creating congestions.

With no extra cost, we have implemented a flow record fragmentation technique that allows to store flow information over multiple records and works as follows:

- The sum of a flow record and of the relative packet records cannot exceed a given amount of memory (e.g. 2Kbytes);
- if when adding a packet record to a flow, the memory limit is exceeded, all recorded information are moved to the main memory and the field “record number” is increased;
- the on-board memory now just contains the flow record with no packets;
- when the flow termination timeout expires the entire record is stored to disk with the Last Record (LR) flag set;

The empirical results show that with multiple records, about 25Mbytes of on board memory is required for an OC-3 monitor with a flow termination timeout of 60s (dashed line in Figure 3). The amount of memory is expected to scale linearly with link speed, thus about 100Mbytes should be needed at OC-12, 400Mbytes at OC-48, and 1.2Gbytes at OC-192.

Note that the bus bandwidth between the capture board and the main memory is not a problem given the compression ratio we can achieve. The use of multiple records also allows a non bursty use of the system bus.

III. Hardware architecture design

Based on the requirements described in Section II, we are prototyping an OC-192 monitor as an extension of the OC-48 monitor better known as the DAG 4 [3]. The above results show that collecting exhaustive packet level data is feasible at OC-192 with the current DAG 4 architecture given that:

- the proposed flow record is used to store packet level

information;
- 66 MHz/64bit PCI bus is used for data transfer between the board memory and the PC main memory;
- at least 1 Gbyte of fast access memory can be made available on the board;
- the use of dedicated hardware for the computing tasks involved in flow encoding.

Recording a flow trace would require the following steps to be done at packet capture time:
- Step 1: capture the packet and process the framing information.
- Step 2: add the 8-byte timestamp and extract the packet fields that will be added to the flow record.
- Step 3: classify the packet, finding what flow it belongs to.
- Step 4: update the flow record if it exists, otherwise create a new one.
- Step 5: when the flow terminates, send the flow record to a host PC along with the associated packet record.

These steps can be easily pipelined and realized with dedicated hardware (e.g. FPGA) so that each step can be done in less than the time between the arrival of two minimum length IP packets at OC-192 speed.

Packet classification is certainly the most expensive and complex step. It requires a one-to-one mapping between a fixed set of packet fields (the 5-tuple made of protocol number and source and destination addresses and port numbers) and the memory portion that contains the flow record. This mapping can be implemented using a hash function and a flow table that contains the addresses of the flow records. A second mechanism (such as a second hash function or a lookup trie) must be also included to solve collision events of the first hashing.

Step 4 instead would require to allocate a fixed amount of memory per flow record. Note that with the use of multiple records the maximum size of a flow record is defined before the trace collection. Therefore, the processor needs to access to the memory address returned by Step 3 and add a packet record (and a new flow record, if the flow is a new one).

Further investigation is needed to evaluate the use of cache memory to store the most recently used flow records and speed up memory accesses. However, it is important to note that for the purposes of trace collection it is also possible to buffer received packets, as long as they have already been timestamped.

IV. Conclusion

We have shown that monitoring high speed links is feasible with current technologies given that we record packet traces in a flow based format. The current technology can scale up to 10Gbps link speed without losing much information on IP packets carried by the link. The same technology could be integrated in routers to provide monitoring facilities on each line card without affecting the router performance.

Providing monitoring facilities on the router backplane, or at speeds higher than 10Gbps, would instead require sampling techniques. Sampling can be done at the packet level, resulting in storing only part of the flow information, or at the flow level, capturing only packets that belong to a subset of the flows.

The main limitation we forsee for very high speed monitoring systems is the memory access speed. As explained earlier, we can not expect the memory access rate to improve at the same speed than processor capacity. Consequently, at backplane throughput, it will be impossible to store information on a per packet basis and either sampling or on-the-fly processing of packet data will be mandatory.

We believe that it is important to work now on monitoring techniques at hundreds of gigabit per second or at terabit per second. By the time these links will become available on operational network, we expect measurement to be an operation critical technology.

We have already described areas where further investigation is needed for example to evaluate the economic feasibility of the proposed monitoring equipment. Moreover, pathological traffic patterns have to be identified in order to design additional mechanisms to guarantee that the capture board can operate in critical scenarios such as SYN flood attacks, port scans or high degree of IP packet fragmentation.

References

[1] C. Fraleigh, C. Diot, B. Lyles, S. Moon, P. Owezarski, D. Papagiannaki, F. Tobagi, *Design and Deployment of a Passive Monitoring Infrastructure*, Passive and Active Measurement Workshop, Amsterdam, April 2001.

[2] J. Cleary, S. Donnelly, I. Graham, A. McGregor, M. Pearson, *Design Principles for Accurate Passive measurement*, Passive and Active Measurement Workshop, Hamilton (New Zealand), April 2000.

[3] The DAG project, http://dag.cs.waikato.ac.nz.

[4] K.C. Claffy, G.C. Polyzos and H.W. Braun, *Application of Sampling Methodologies to Network Traffic Characterization*, ACM Sigcomm 1993

[5] K.C. Claffy, H.W. Braun, G.C. Polyzos, *A parameterizable methodology for Internet traffic flow profiling*, IEEE JSAC 1997

[6] B. Ryu, D. Cheney, H. Braun, *Internet Flow Characterization: Adaptive Timeout Strategy and Statistical Modeling*, Passive and Active Measurement Workshop, Amsterdam, April 2001.