Packet Stats Design Requirements

Author: Jad Naous jnaous@stanford.edu

1.0 Goal

The goal of the project is to produce a simple design that can be incorporated into a switch to collect fairly accurate statistics on arrival/departure/drop times of packets on output queues. The data is aggregated into a raw Ethernet packet which is then forwarded to a specified MAC address or broadcast to be processed further.

2.0 Specifications

This section lists the requirements in detail.

2.1 Definitions

Event: a record of a packet arrival/departure/drop consisting of the event time and packet length, and any other information necessary to process the event.

User: the host CPU performing operations on behalf of an administrator

Time difference: Time elapsed since last event

Timestamp: Time elapsed since last timer reset

2.2 Statistics Capture and Timing

1- The design shall be able to capture all events on all output queues in the switch.
2- The user shall be able to set which output queues to monitor
3- The user shall be able to reset the timer
4- Timer resolution: The user shall be able to set the timer resolution/precision as (2^c)*8ns where ‘c’ can be 0 to 7 (i.e. resolutions from 8 ns to 1.024us)
5- Packet stats capture shall be able to run a very long time (years) without any intervention from the user
6- The user shall be able to see the number of events that were missed/could not be recorded.

2.3 Events Packet and Transmission

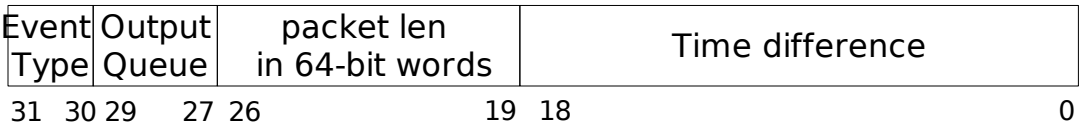
1- The captured statistics shall be written into a UDP packet

2- The user shall be able to program the packet's Ethernet, IP and UDP headers
3- The packet shall be self contained, needing no information from another packet
4- The user shall be able to select the port(s) to send the events packets from
5- The user shall be able to force the packet to be transmitted even if it is not complete

2.4 Events Format

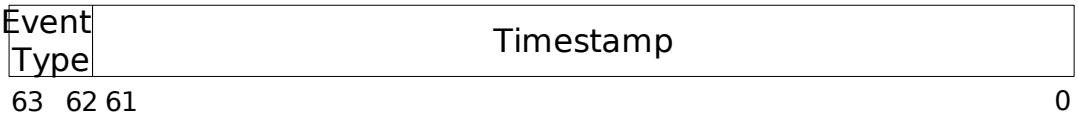
There will be 4 event types:

1- Arrival/Departure/Drop (short) Events will have the following format



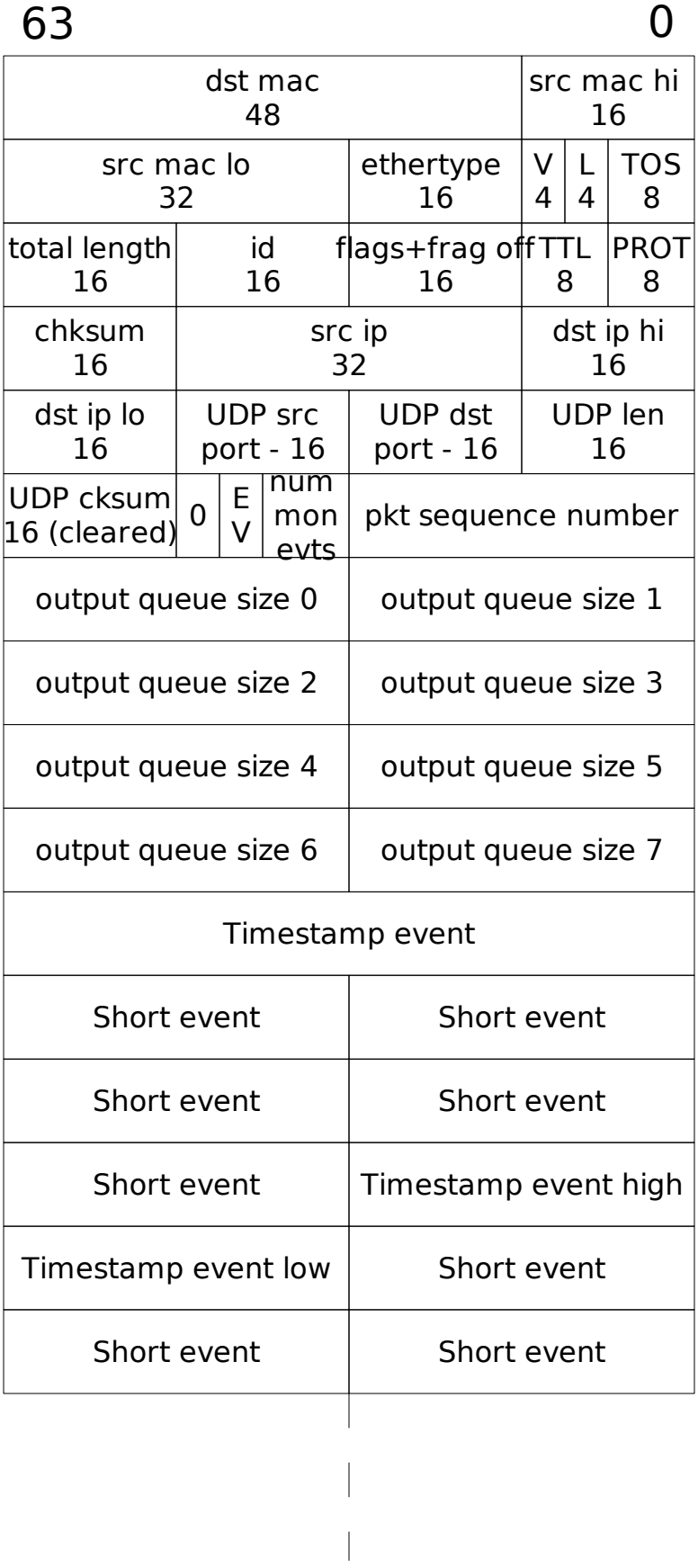
In the case of 128ns timer resolution, this format will allow time differences of 52.4 ms before wrap-around. Also note that the packet length here is in words of 64-bits.

2- Timestamp Event: used to maintain independence between packets and to solve the Time Difference wrap-around problem



2.5 Packet Format

Following the Ethernet, IP, and UDP headers, there are 4 reserved bits, then 4 bits indicating the version of the event packet capture system, then 8 bits indicating the number of event types excluding the Timestamp event type. This is followed by a 32 bit sequence number (starting at 0) and a list of the queue sizes before the first short event. A series of short events then follows. If the time difference since the last event is longer than 19 bits, a new Timestamp event is inserted.


63
0
dst mac 48
src mac hi 16
src mac lo 32
ethertype 16
V 4
L 4
TOS 8
total length 16
id 16
flags+frag off 16
TTL 8
PROT 8
chksum 16
src ip 32
dst ip hi 16
dst ip lo 16
UDP src port - 16
UDP dst port - 16
UDP len 16
UDP cksum 16 (cleared)
0
E V
num mon evts
pkt sequence number
output queue size 0
output queue size 1
output queue size 2
output queue size 3
output queue size 4
output queue size 5
output queue size 6
output queue size 7
Timestamp event
Short event
Short event
Short event
Short event
Short event
Timestamp event high
Timestamp event low
Short event
Short event
Short event

2.6 Bandwidth requirements

A single 1500 bytes events packet will have enough space for 1 timestamp event and ~350 short events. If there are 2 events every 1 us for every queue, then every events packet will have enough space for around 184 us per queue. If all queues are being monitored, then every events packet will have around 46 us worth of statistics.

In the single queue case, the bandwidth requirement is 1500*8/184us=65.2 Mbps. When monitoring all queues, the bandwidth requirement is 261 Mbps on average. Monitoring all queues can stress the links when there is a lot of activity.