tcp_test_20170721

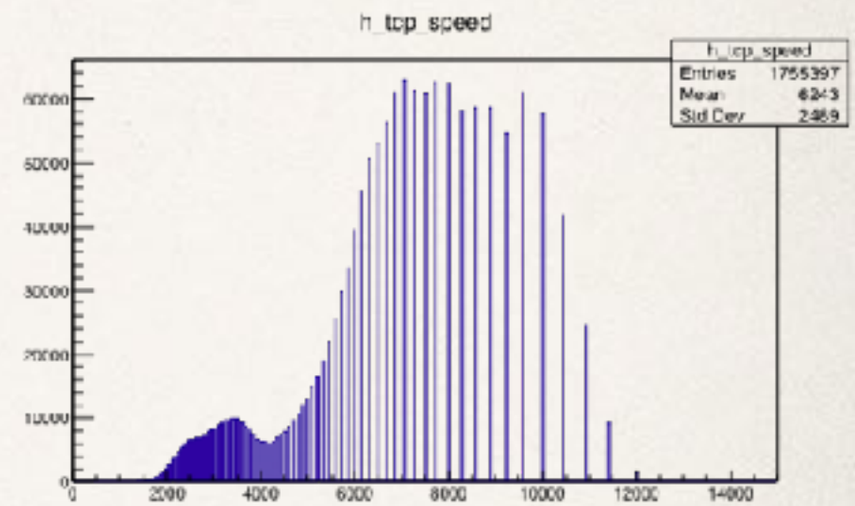
y.ma@riken.jp

2017/07/21

Outline: further investigations based on the feedback

- mini_daq bug fixes: spike & memcpy bugs
- iperf+UDP for data loss/data collision study
- Use jumbo frame for TCP;
- TCP congestion control algorithm

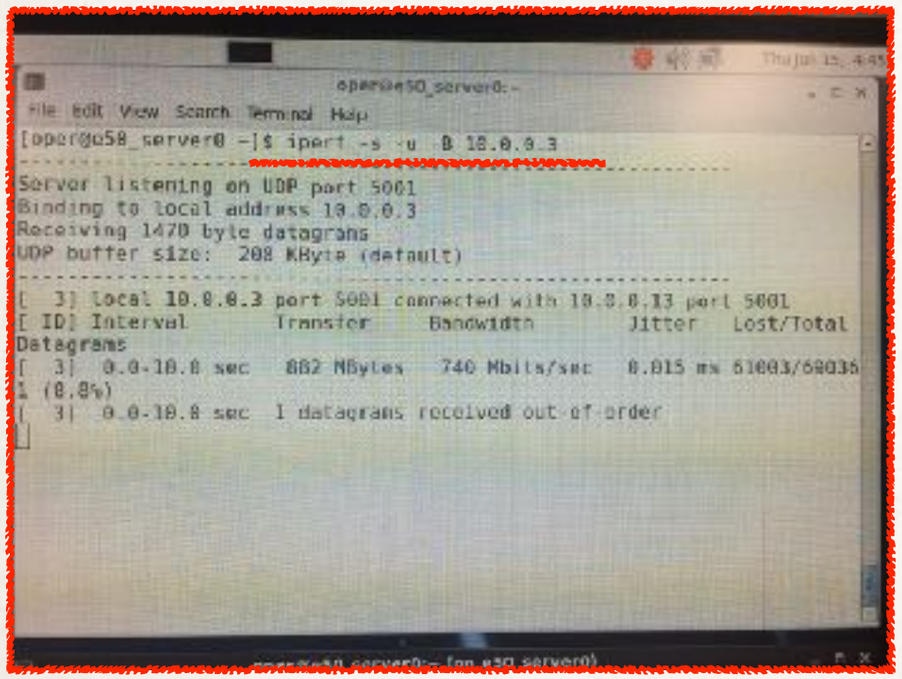
mini_daq bug fixes



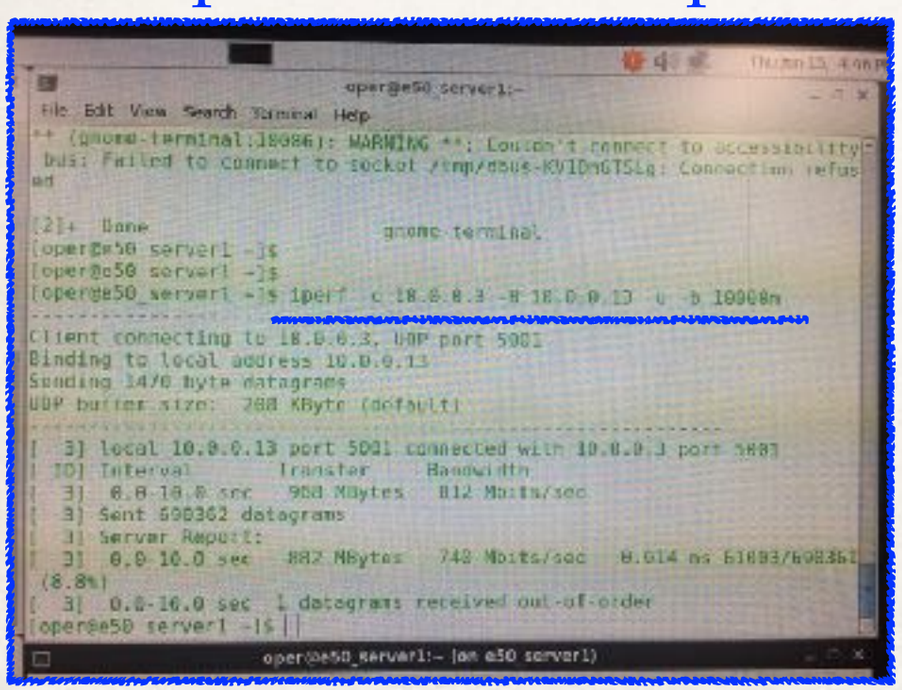
- Online analyzer modification:
  - Use nano second time stamp to “cure” the spikes
  - Use recorder_thrd.c to fill TCP speed histogram to avoid local ethernet throughput
  - Handling dynamic data_length

iperf+UDP for data loss/data collision study

iperf server setup



iperf client setup



Band width parameter seems to be most important:

1, by setting “-b 10000m”, the data loss is ~10% no matter four pair connections or single connection

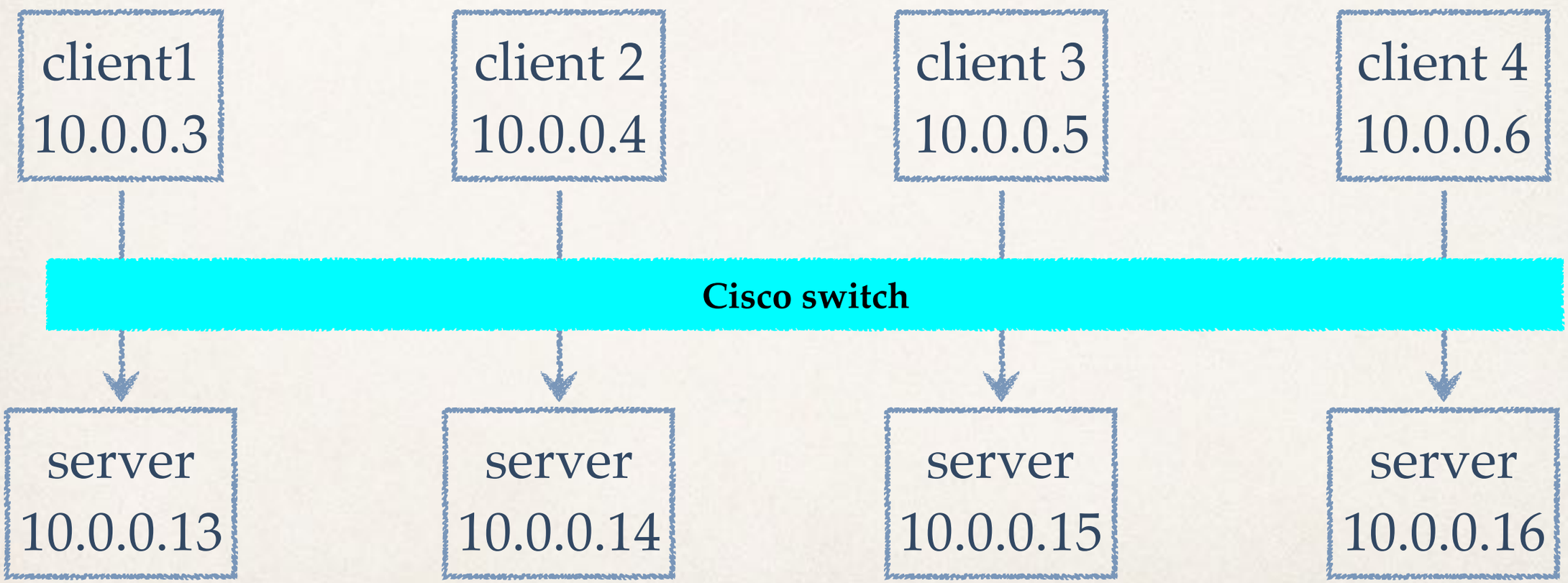
2, by setting “-b 1000m”, the data loss is ~3% no matter four pair connections or single connection

conclusion: *the current iperf version doesn’t support 10gbps??*

(iperf version 2.0.5)

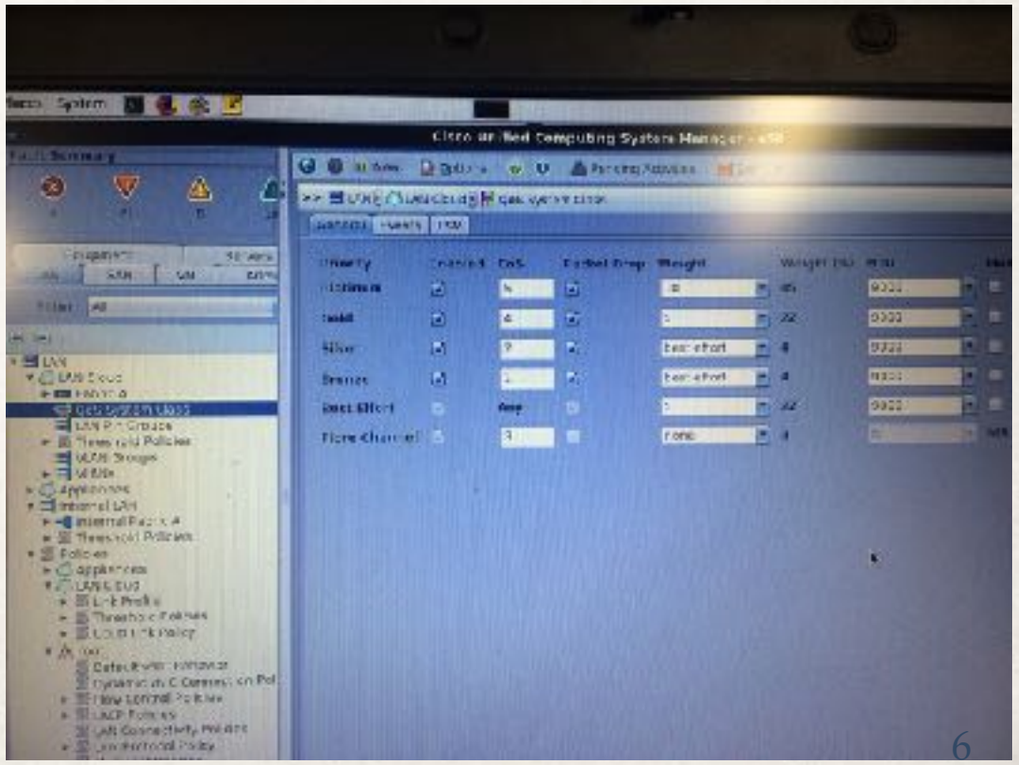
Test configuration

Use the following configuration for test



Select Jumbo fram (9000Bytes/frame)

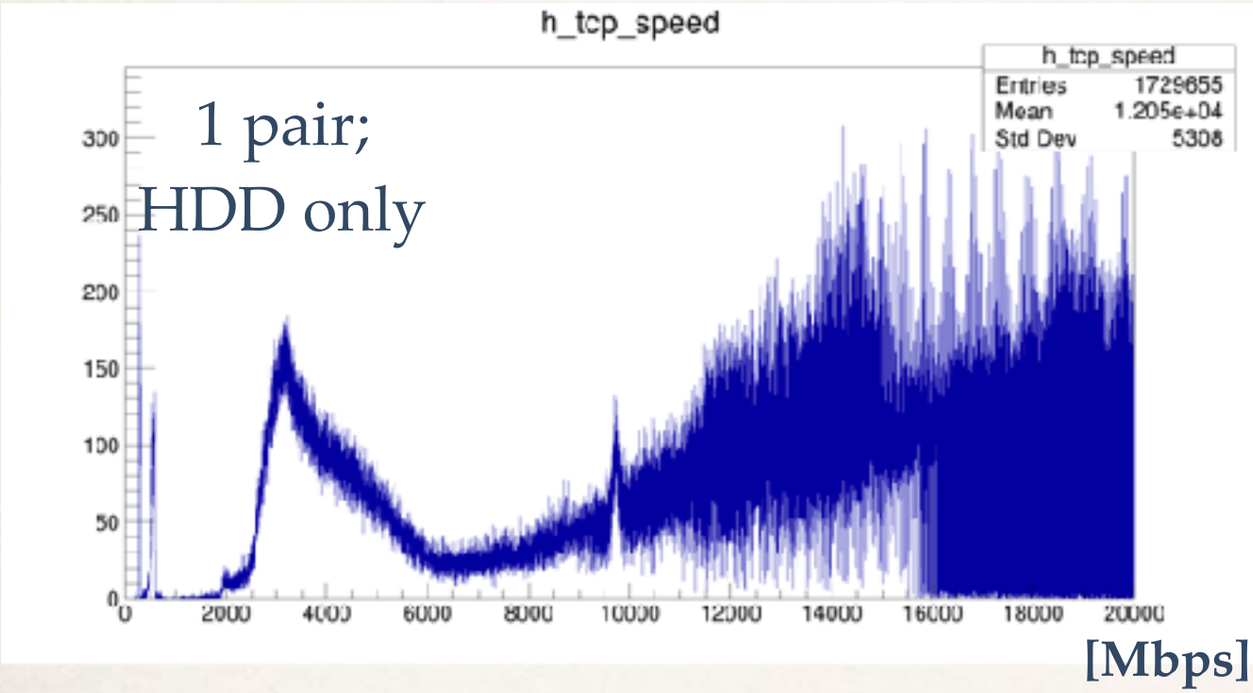
- From E50 server0 and server1, select Jumbo frame with
  [root@e50_server0 oper]# ifconfig ens6f3 mtu 9000
- Confirm the change with [root@e50_server0 oper]# ifconfig ens6f3
- Also configure Cisco UCS 6120 for Jumbo frame



TCP congestion algorithm

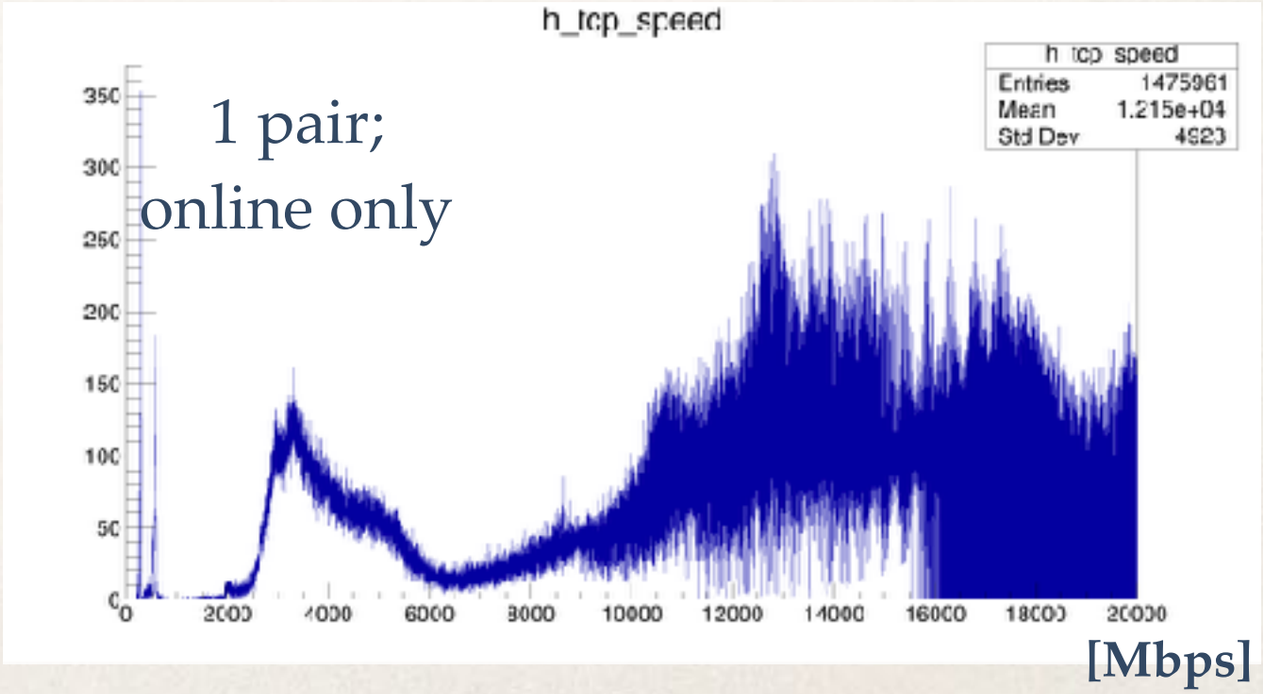
- Check available module: ls /lib/modules/`uname -r`/kernel/net/ipv4/
- Load module: /sbin/modprobe tcp_htcp
- To check the default congestion algorithm: sysctl net.ipv4.tcp_congestion_control
  - results obtained so far are based on default “cubic” algorithm
- To check the control algorithm allowed: sysctl net.ipv4.tcp_allowed_congestion_control
- To set the control algorithm: sysctl -w net.ipv4.tcp_congestion_control=reno

Results

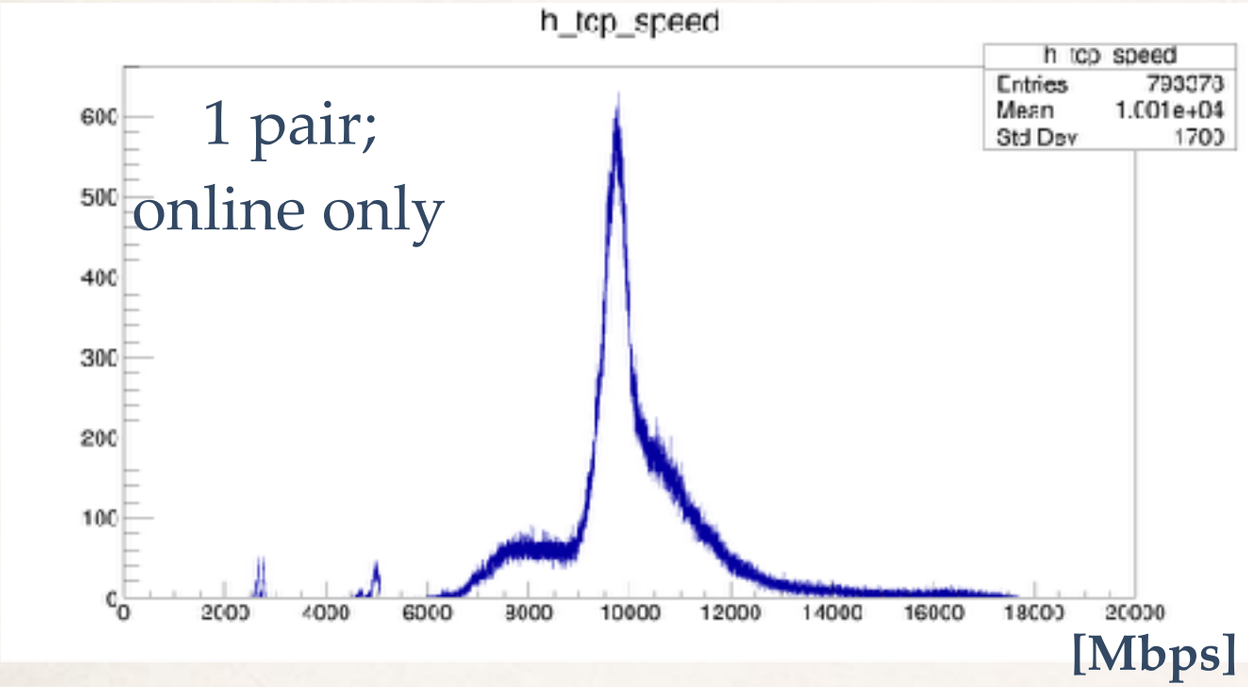


packet = 30kB, 10k buffer, Jumbo frame, congestion control = “highspeed”

No serious overhead found; use online histogram to evaluate TCP speed

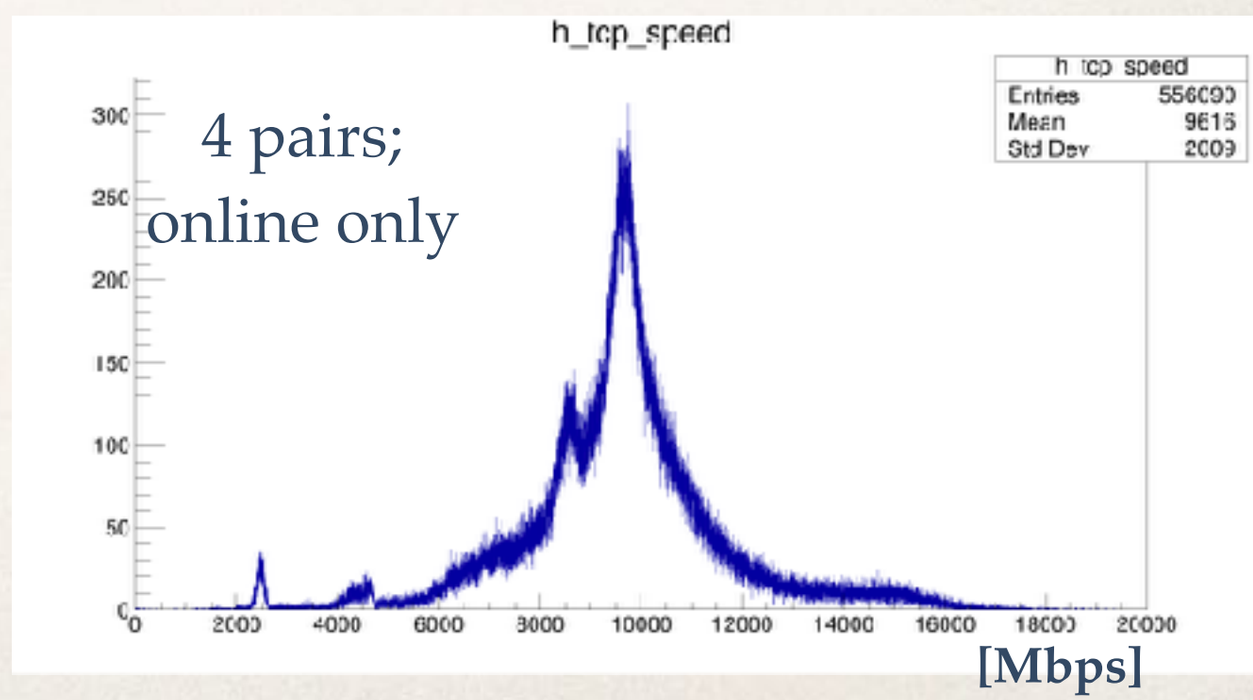


Results

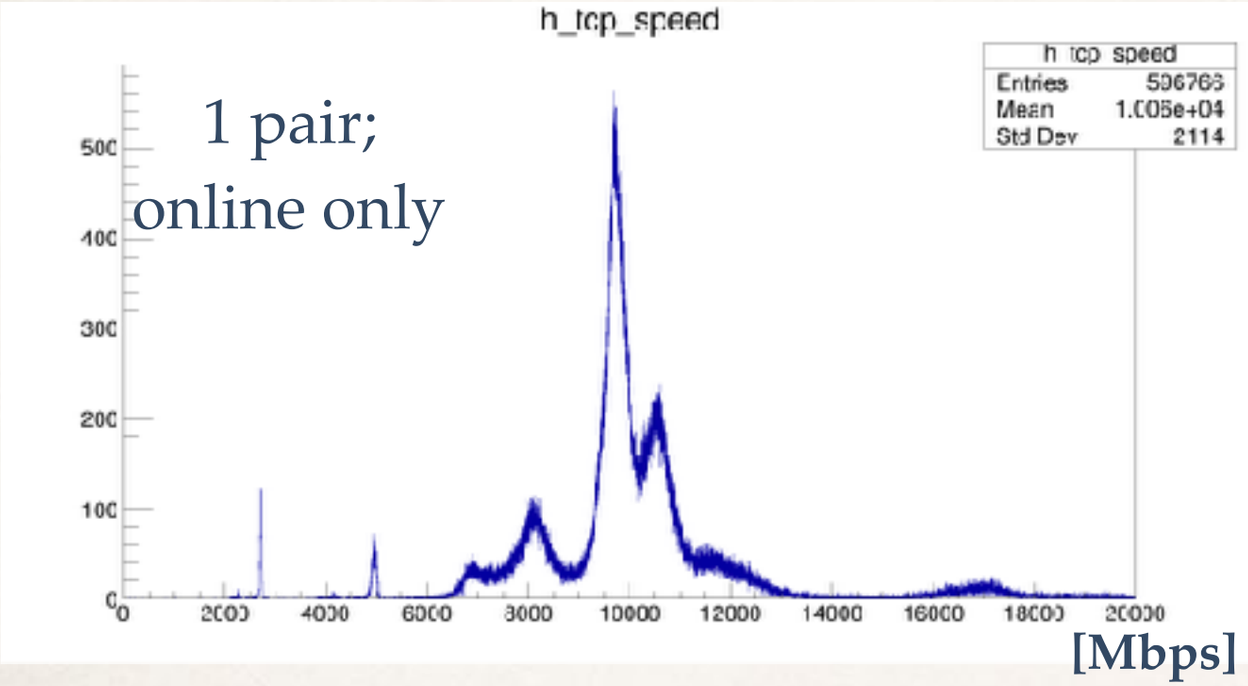


packet = 300kB, 10k buffer, Jumbo frame, congestion control = “highspeed”

performance converged; use two Gauss for P.D.F?

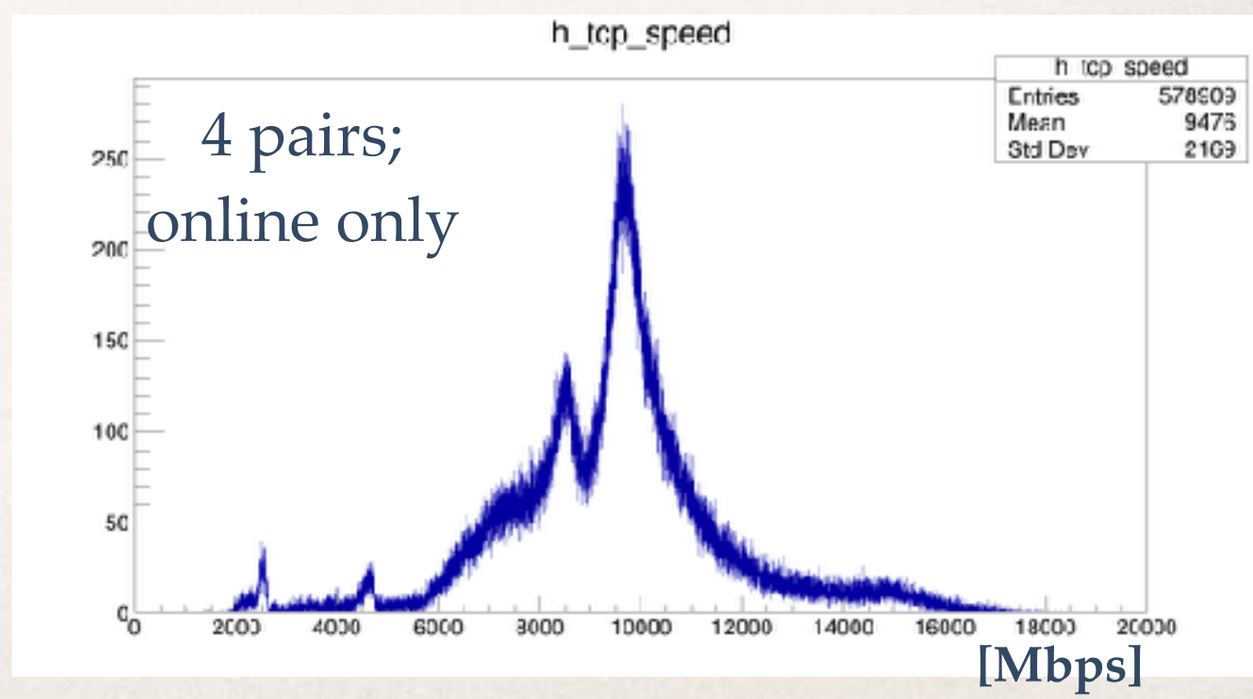


Results

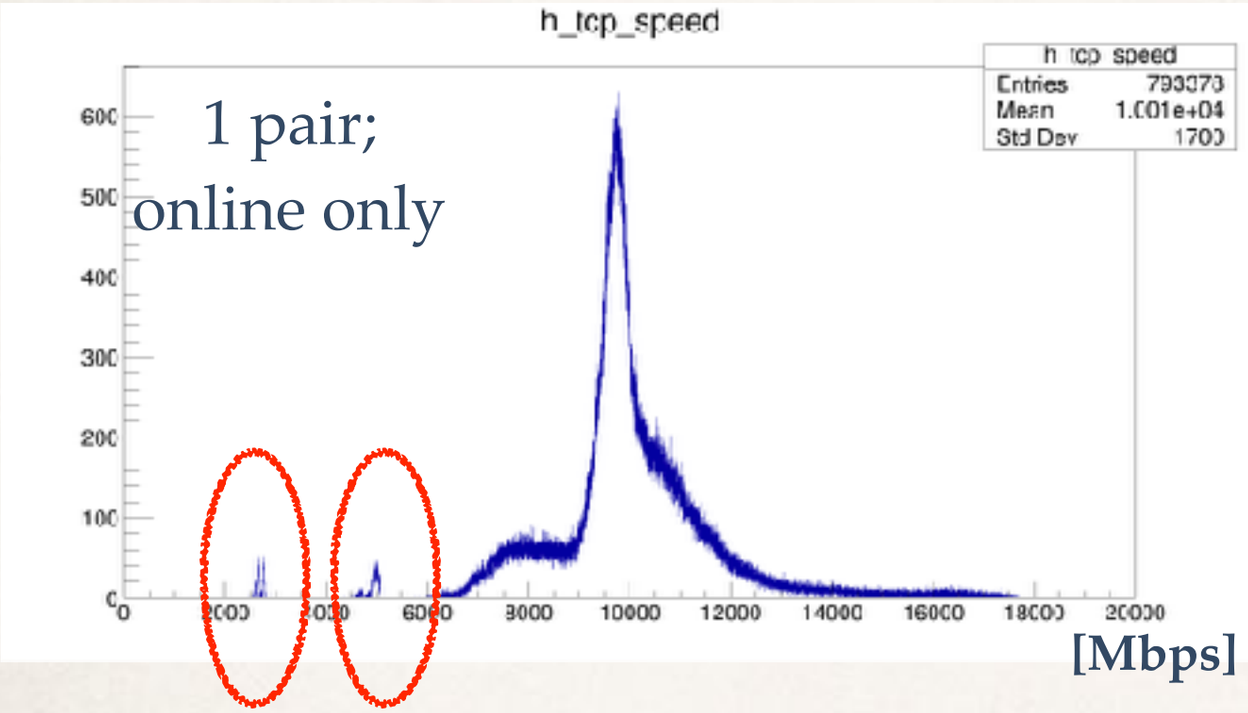


packet = 300kB, 10k buffer, Jumbo frame, congestion control = “cubic”

performance converged; use two Gauss for P.D.F?

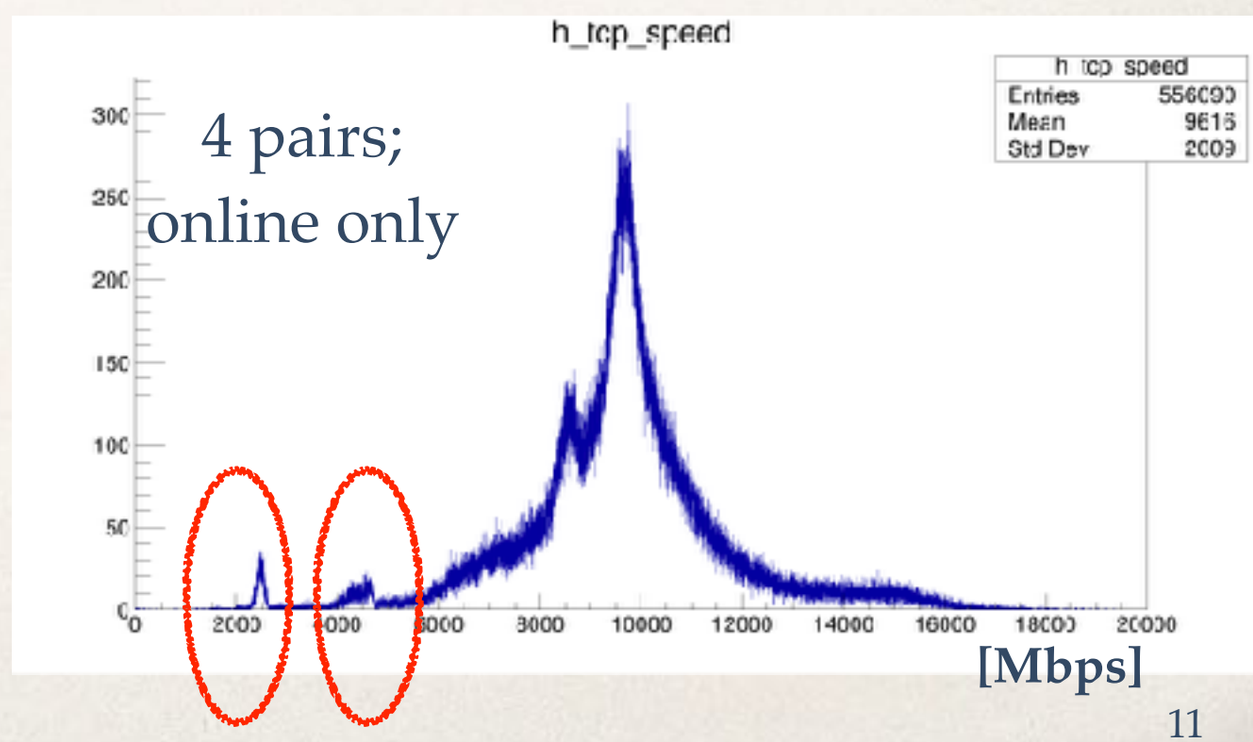


One more thing…

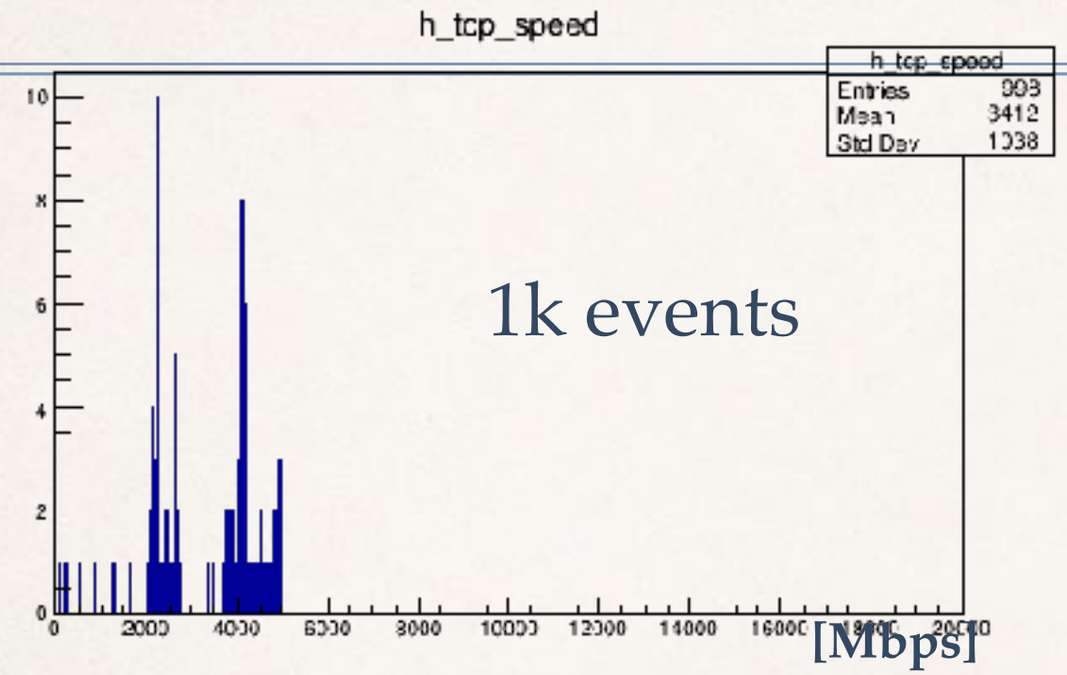


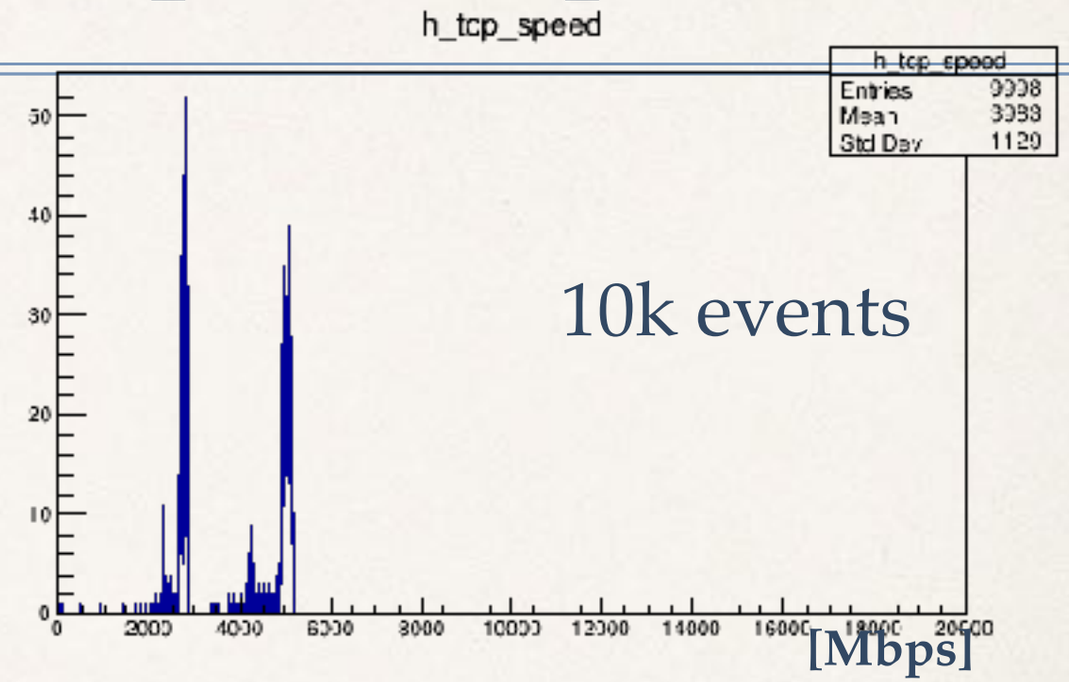
packet = 300kB, 10k buffer, Jumbo frame, congestion control = “highspeed”

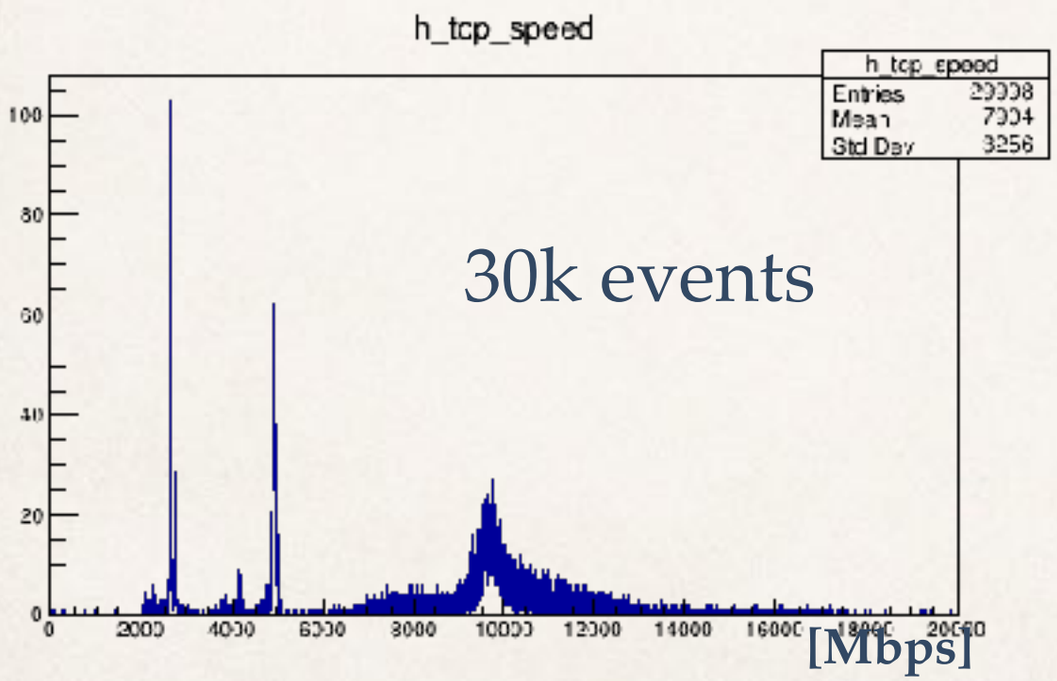
What are these structures??

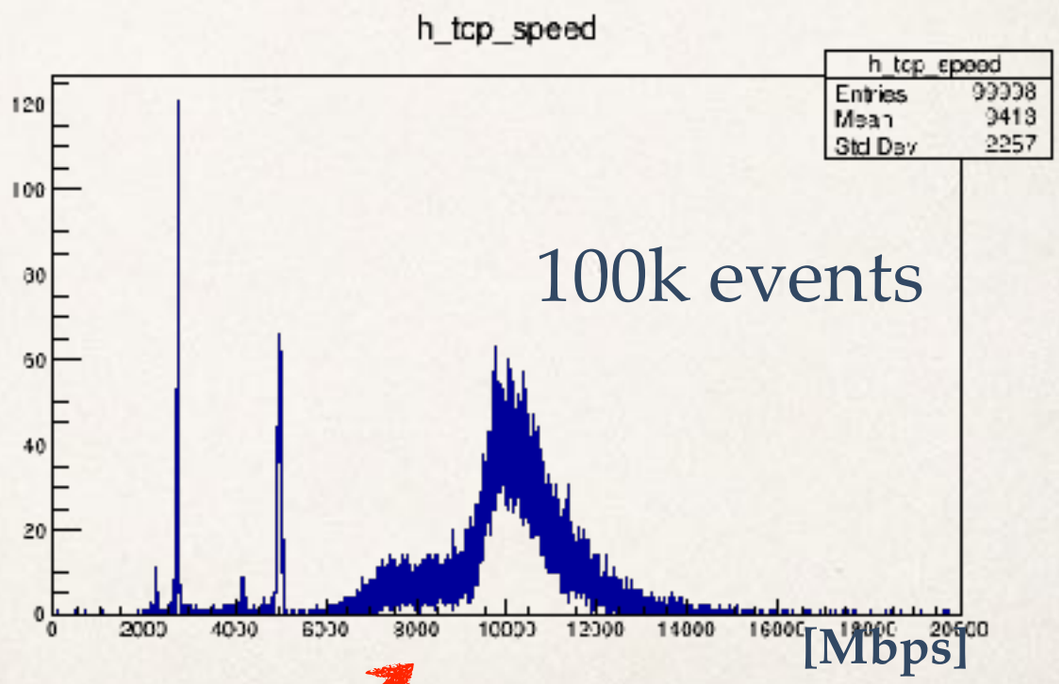


TCP/IP needs time to speed up…



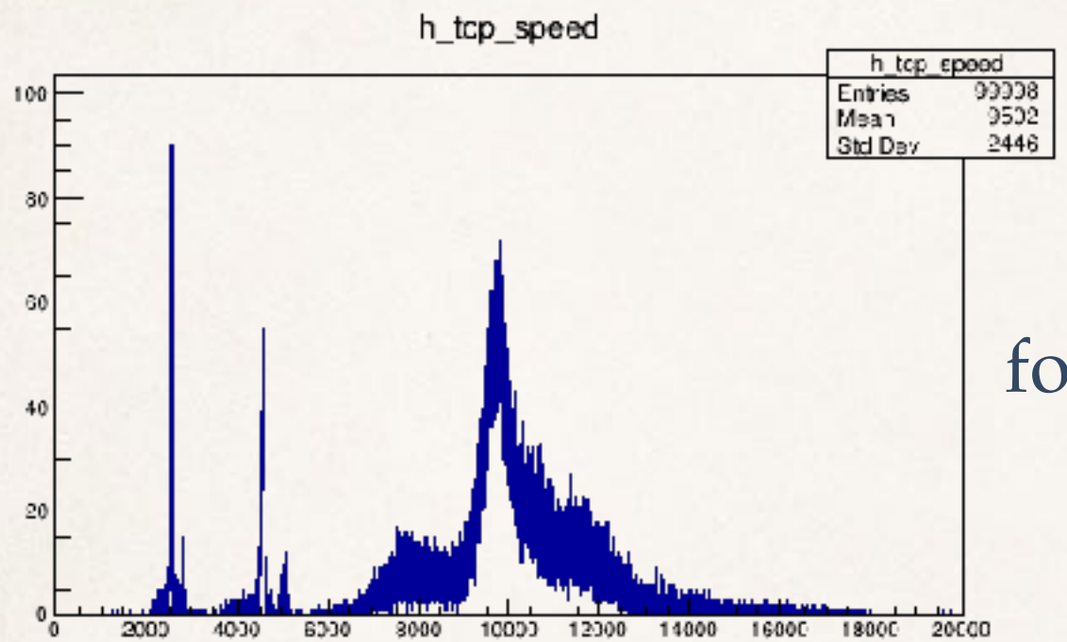






low speed events saturated and high speed events increase

TCP/IP needs time to speed up…



100k events;
force to sleep(1) every 1k events;
performance doesn't change;

- TCP/IP needs ~30k cycles to achieve max speed of 10gbps;
- long dead time/need more buffer in the initial stage;
- after speeds up, force to sleep doesn't change the performance;
- what's the reason? seems not from system scheduling…

Summary & todo

- Updated TCP speed histograms provide more reliable information
- Jumbo frame slightly improves the performance
- Congestion algorithm seems not very effective
- Packet size is most critical for a good performance: accumulate ~300kB before sending to TCP buffer
- Use two Gauss distribution to represent TCP P.D.F?
  - one for Linux timestamp resolution; another for TCP speed fluctuation??
- *TCP/IP needs time to speed up??*

- Backup

Backup: modification in 10.0.0.5/builder_thrd.c

```
/* to check tcp speed change as time */
if ( server_event->server_header.server_event_id >= 100000 )
    {
        state_register->daq_register = QUIT;
    }
/* to check if tcp speed will be dropped after sleep */
if ( server_event->server_header.server_event_id %1000 == 0 )
    {
        printf("sleeping now ... \n");
        fflush(stdout);
        sleep(1);
    }
/* finish the check */
```

- use Epson as DHCP server for Cisco6120?
- smart routing—>performance degraded?

- Bigger buffer? kernel TCP tuning?
- RDMA: transport mechanism behave like DMA? less CPU consumption; RoCE3 UDP
- compare TCP with RDMA and UDP
- traffic pattern of all to one "*simultaneously*"
  - one more pattern? four client on the same server socket simultaneously instead of looping?