Proposed Definitions for Imix and Latency Histogram

The following definitions for Imix and Latency histogram specification in Traffic configuration in VSPERF were proposed and agreed during the months of February-March 2018.

```json
'imix': {  
'enabled': False,  
'type': 'proportions', # 'proportions' or 'genome'  
'proportions': [  
# This is required to support custom sizes  
{'pkt_size': 64, 'proportion': 60.0},  
{'pkt_size': 594, 'proportion': 30.0},  
{'pkt_size': 1518, 'proportion': 10.0},  
],  
'genome': 'aaaaaadddg',
},

# If the enabled is 'True', for the latency histogram or percentiles,  
# and if the TGen does not support, then, no error is returned.  
# Those values will not be seen in the results.

'latency_histogram': {  
'enabled': False,  
'bucket_limits_us': [1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000, 11000, 12000, 13000, 14000, 15000, 16000],
},

'latency_percentiles': {  
'enabled': False,  
'percentiles': [50.0, 75.0, 90.0, 99.0, 99.9, 99.99],
},
```

Additional Information:

'flow-control' flag: This parameter refers to IEEE 802.3x standard. This method is based on 'paused-frame' - which halts the transmission of the sender for a period of time. The pause-time varies from 0000 - FFFF (a 2-byte value). The pause-time values are computed by Tx and Rx appropriately. The pause-times are specified in unit of slot-times, which is 512 bit times for <= 100Mbps link. So, for 100Mbps with Pause-time as 2, would translate to 10.24 us. This method applies to link-basis and not per-flow basis.

Input1: This definition of frame/packet content is not flexible enough to cover future extensions. It might be good to introduce a new way (at least as an option) of frame/packet content definition, e.g. in form of scapy definition. It will allow user to configure huge variety of network protocols without any changes to vsperf itself. Of course, this will work only for some traffic generators (e.g. T-Rex), which are able to accept a scapy template. Currently TRAFFIC dictionary covers all options somehow related to the traffic, which should be passed through DUT. Of course only part of these options specify the content of frames/packets, which are sent by traffic generator. May be, we should not move non-frame related options out of the TRAFFIC array, but to create a new FRAME/PACKET related dictionary, which will hold only options related to traffic "content".

Input2: The scapy part will only give the general format of the frame, the flow spec will dictate what goes into the addressing fields of the formed frame. Just looking at how flows can be formed: many fields (MAC, IP, port) can change to from different flow profiles which can have dramatic impact on the benchmark results (even on the same SUT). This is a critical part of any benchmark that we need to standardize (i.e. define exactly how the flows are defined). In NFVbench we also need to split flows into subgroups so they can be sent to different service chains, and this requires setting up the dest fields properly.

Additional Clarifications

Q: Which benchmark makes use of the 'flow_control' feature? Trex has the concept of inter-stream gap (gap between 2 streams) and inter-burst gap (gap between 2 bursts inside a stream).

A. This feature is used to define dependencies among various options and to effectively write step driven testcases. In case that configuration files would use e.g. yaml format, it would be required to heavily modify vsperf code to achieve the same functionality. flow control is supported by DPDK and OVS with DPDK support. There are two testcases in vsperf to verify flow control functionality: ovsdpdk_flow_ctrl_rx and ovsdpdk_flow_ctrl_rx_dynamic