Which Standard for Ethernet Statistics?

THOMAS MONJALON - MELLANOX
Why Statistics?

- Monitoring
- Problem detection
- Debugging
- Performance analysis
Current DPDK Implementation

WHAT ARE THE LIMITATIONS?
Basic Stats Definitions

- Rx/Tx Packets
  - including errors between PHY and CPU?
  - Rx Missed, Rx Errors, Tx Errors
  - Rx mbuf allocation failures = CPU / SW issue
- Rx/Tx Bytes
  - including errors between PHY and CPU?
  - including CRC?
- Per-Queue Statistics
  - Rx/Tx Packets
  - Rx/Tx Bytes
  - Rx Errors (including Missed?)
  - no Tx Errors counter
  - no Rx mbuf allocation failures
Stats per Queue Mapping

- **Maximum Queues number**
  - DPDK build-time defined: `RTE_ETHDEV_QUEUE_STAT_CNTRS`
  - Default: 16

- **Function to Map Counters with Queues**
  - `rte_eth_dev_set_[rt]x_queue_stats_mapping()`
  - relevant only for ixgbe which is limited in counters
Adding More in Basic Stats?

• More fields?
• More queue counters?

Drawbacks
• Where is the limit?
• Memory usage
• Performance of big query
• May performance of stats query be a concern?
Extended Stats

rte_eth_xstats_*( )

- **Name / Id / Value**
  - 1:1 mapping between string name and **64-bit id**
  - Value = unsigned 64 bits

- **Query all or by id**

- **Basic stats are exposed also as xstats**
  - rx_good_packets / tx_good_packets
  - rx_good_bytes / tx_good_bytes
  - rx_errors / tx_errors
  - rx_missed_errors
  - rx_mbuf_allocation_errors
  - rx_qXpackets / tx_qXpackets
  - rx_qXbytes / tx_qXbytes
  - rx_qXerrors
xstats Naming Scheme

- Naming scheme is defined in doc
  - [http://doc.dpdk.org/guides/prog_guide/poll_mode_drv.html#scheme-for-human-readable-names](http://doc.dpdk.org/guides/prog_guide/poll_mode_drv.html#scheme-for-human-readable-names)
  - Fields separated with underscore
    - direction (rx / tx)
    - detail 1 (can be queue number)
    - detail 2
    - detail n...
    - unit (packets / bytes)

- Current implementation of basic stats per queue not compliant
  - "rx_q%u%s" misses an underscore: "rx_q%u_%s"
  - API break?
No xstats Definitions

- xstats are inherited from driver-specific counters
- xstats names are not standardized
- xstats ids can be different per port

- xstats should include standardized basic counters
- **Reserve ids** for what is considered basic
- **Precisely define** meaning of each basic stat
xstats Query

- Can query all xstats
  - `rte_eth_xstats_get()`

- Can query a subset of xstats
  - `rte_eth_xstats_get_by_id()`

- Reserved ids = no need of name query = fast subset query

- No reserved ids for stats per queue?
xstats Id Scheme

- First 256 ids reserved for well-known **basic** stats
- Second part available for custom **driver-specific** stats

- Reserve low ids for **port**-stats
  - Space = 24 bits

- Reserve high ids for **queue**-stats
  - Reserve 64K stats per queue
  - Reserve 16M queues
  - Total = 40 bits
  - **Breaks API** assumption: id ≠ array index
Deprecate Legacy Stats?

- No breakage of legacy basic stats
- New definitions apply **only** to xstats

- Legacy stats per queue can be removed from `rte_eth_stats` in future
- `rte_eth_stats` can be deprecated in future
New Definitions
Counter Size

**In Papers**
- Simple Counter = 32 bits
- High Capacity Counter = 64 bits

**In DPDK (current and future)**
- Counter = **64 bits**
  - 23 years counting bytes at 200 Gbps
Multiple Standards

- Interfaces Group (**IF-MIB**): RFC 2863
- Broadband Forum: TR-181
  - inspired by IF-MIB
- IEEE 802.3 Ethernet Working Group
- Ethernet-like Interface Types (**EtherLike-MIB**): RFC 3635
  - based on 802.3 and IF-MIB
- Remote Network Monitoring (**RMON1-MIB**): RFC 2819
  - no Rx/Tx
- Remote Network Monitoring for High Capacity (**HCRMON-MIB**): RFC 3273
  - high capacity counter (64-bit) + overflow counter (32-bit)
HOW STANDARDS PROLIFERATE:
(SEEN: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)

SITUATION: THERE ARE 14 COMPETING STANDARDS.

14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES. YEAH!

SOON:

SITUATION: THERE ARE 15 COMPETING STANDARDS.

http://xkcd.com/927
Multiple Implementations

- SNMP
- Linux netdev
  - ethtool ≈ xstats
- OVS
- DPDK
- All other Operating Systems and Networking Libraries...
Representation of Not Available

- Initialize all stats to $\text{UINT64\_MAX} = \text{N/A}$
- Reset **supported** stats to 0
Counting Bytes including CRC?

- **DPDK**: depends on driver?
  - Should not depend on CRC stripping configuration
- **Linux**: no?
- **IF-MIB**: yes
- **EtherLike-MIB**: yes, and count only valid packets
- **RMON1-MIB**: yes

- Note: virtual links have no CRC
Rx total packets/bytes

- **PHY view**: including errors from PHY to CPU
- **CPU view**: only good packets received by application

- **DPDK**: depends on driver
- **Linux**: depends on driver
- **IF-MIB**: PHY view, only bytes
- **TR-181**: PHY view
- **EtherLike-MIB**: PHY view, only bytes of valid packets
- **RMON1-MIB**: PHY view
**Tx total packets/bytes**

- **PHY view**: not including errors from CPU to PHY
- **CPU view**: all packets accepted by the API

- **DPDK**: depends on driver
- **Linux**: depends on driver
- **IF-MIB**: PHY view, only bytes
- **TR-181**: PHY view
- **EtherLike-MIB**: PHY view, only bytes of valid packets
- **RMON1-MIB**: CPU view

- If TSO?
- If offload not possible?
Rx good packets/bytes

- CPU view: received by application
- DPDK: no
- Linux: no
- IF-MIB: no, but = unicast + multicast + broadcast
- RMON1-MIB: no
Tx good packets/bytes

- PHY view: sent on the link
- DPDK: no
- Linux: no
- IF-MIB: no, but = unicast + multicast + broadcast - errors - discards
- RMON1-MIB: no
Rx/Tx per size

- **DPDK**: xstats driver-specific
- **Linux**: ethtool driver-specific
- **OVS**: yes, [1024-1522], [1523-max]
- **IF-MIB**: yes
- **EtherLike-MIB**: no
- **RMON1-MIB**: no

**RFC 2819**

- 64
- 65 - 127
- 128 - 255
- 256 - 511
- 512 - 1023
- 1024 - 1518

- common last range: [1024-max]
Rx/Tx unicast

- DPDK basic: no
- Linux netdev: no
- IF-MIB: yes, CPU view
- RMON1-MIB: no
Rx/Tx multicast

- DPDK basic: no
- Linux netdev: yes, Rx
- IF-MIB: yes, CPU view
- RMON1-MIB: yes, good packets only
Rx/Tx broadcast

- DPDK basic: no
- Linux netdev: no
- IF-MIB: yes, CPU view
- RMON1-MIB: yes, good packets only
Rx/Tx pause frames

- DPDK basic: no
- Linux netdev: no
- IF-MIB: no
- EtherLike-MIB: yes
- RMON1-MIB: no
Rx errors total

- **DPDK**: yes, all but nobuf
- **Linux netdev**: yes, all but nobuf
- **IF-MIB**: yes, all but nobuf + missed
- **EtherLike-MIB**: yes = alignment + CRC + oversize + internal MAC
- **RMON1-MIB**: no
Rx buffer allocation failure

- CPU / SW side
- **DPDK**: yes
- **Linux netdev**: yes, Rx dropped
- IF-MIB: no
- RMON1-MIB: no
Rx missed

- **DPDK**: yes
- **Linux netdev**: yes + FIFO errors
- **IF-MIB**: yes, discards
- **RMON1-MIB**: no
Rx under/oversize

- DPDK basic: no
- **Linux netdev**: yes, rx_length_errors + rx_over_errors
- IF-MIB: no
- **EtherLike-MIB**: yes, only oversize
- **RMON1-MIB**: yes, out of [64-1518]
  - fragments = undersize with error
  - jabbers = oversize with error
Rx CRC errors

- DPDK basic: no
- Linux netdev: yes
- IF-MIB: no
- EtherLike-MIB: yes
- RMON1-MIB: yes, merged with alignment errors
Rx alignment errors

- DPDK basic: no
- Linux netdev: yes, frame errors
- IF-MIB: no
- EtherLike-MIB: yes
- RMON1-MIB: yes, merged with CRC errors

- Is there a need?
Rx unsupported protocol

- DPDK basic: no
- Linux netdev: no
- IF-MIB: yes
- RMON1-MIB: no

- Is there a need?
Tx errors total

- DPDK: yes
- Linux netdev: yes
- IF-MIB: yes, all but discarded
- EtherLike-MIB: yes = SQE test + collisions + internal MAC + carrier sense
- RMON1-MIB: no
Tx discards

- DPDK basic: no
- Linux netdev: yes, dropped
- IF-MIB: yes
- RMON1-MIB: no
Tx FIFO errors

- DPDK basic: no
- Linux netdev: yes, overrun
- IF-MIB: no
- RMON1-MIB: no
Tx carrier errors

- DPDK basic: no
- Linux netdev: yes, e.g. link down
- IF-MIB: no
- EtherLike-MIB: yes
- RMON1-MIB: no
Collisions

- DPDK basic: no
- Linux netdev: yes
  - Tx aborted
  - Tx window errors = late collisions
- IF-MIB: no
- EtherLike-MIB: yes
- RMON1-MIB: yes
Conclusion

TODO for 20.11 (will break API)

Add new definitions as reserved xstats ids for well-known basic statistics needs.

Deprecate legacy basic statistics.
Questions?

Volunteers?