NAME

MQPRIO – Multiqueue Priority Qdisc (Offloaded Hardware QOS)

SYNOPSIS

tc qdisc ... dev dev (**parent** classid | **root**) **[handle** major: **] mqprio [num_tc** tcs **] [map** P0 P1 P2... **] [queues** count1@offset1 count2@offset2 ... **] [hw** 1|0 **] [mode** dcb|channel] **] [shaper** dcb| **[bw_rlimit min_rate** min_rate1 min_rate2 ... **max_rate** max_rate1 max_rate2 ... **]]**

DESCRIPTION

The MQPRIO qdisc is a simple queuing discipline that allows mapping traffic flows to hardware queue ranges using priorities and a configurable priority to traffic class mapping. A traffic class in this context is a set of contiguous qdisc classes which map 1:1 to a set of hardware exposed queues.

By default the qdisc allocates a pfifo qdisc (packet limited first in, first out queue) per TX queue exposed by the lower layer device. Other queuing disciplines may be added subsequently. Packets are enqueued using the **map** parameter and hashed across the indicated queues in the **offset** and **count.** By default these parameters are configured by the hardware driver to match the hardware QOS structures.

Channel mode supports full offload of the mqprio options, the traffic classes, the queue configurations and QOS attributes to the hardware. Enabled hardware can provide hardware QOS with the ability to steer traffic flows to designated traffic classes provided by this qdisc. Hardware based QOS is configured using the **shaper** parameter. **bw_rlimit** with minimum and maximum bandwidth rates can be used for setting transmission rates on each traffic class. Also further qdiscs may be added to the classes of MQPRIO to create more complex configurations.

ALGORITHM

On creation with 'tc qdisc add', eight traffic classes are created mapping priorities 0..7 to traffic classes 0..7 and priorities greater than 7 to traffic class 0. This requires base driver support and the creation will fail on devices that do not support hardware QOS schemes.

These defaults can be overridden using the qdisc parameters. Providing the 'hw 0' flag allows software to run without hardware coordination.

If hardware coordination is being used and arguments are provided that the hardware can not support then an error is returned. For many users hardware defaults should work reasonably well.

As one specific example numerous Ethernet cards support the 802.1Q link strict priority transmission selection algorithm (TSA). MQPRIO enabled hardware in conjunction with the classification methods below can provide hardware offloaded support for this TSA.

CLASSIFICATION

Multiple methods are available to set the SKB priority which MQPRIO uses to select which traffic class to enqueue the packet.

From user space
: A process with sufficient privileges can encode the destination class directly with SO_PRIORITY, see socket(7).

with iptables/nftables
: An iptables/nftables rule can be created to match traffic flows and set the priority. iptables(8)

with net_prio cgroups
: The net_prio cgroup can be used to set the priority of all sockets belong to an application. See kernel and cgroup documentation for details.

QDISC PARAMETERS

num_tc Number of traffic classes to use. Up to 16 classes supported.

map The priority to traffic class map. Maps priorities 0..15 to a specified traffic class.

queues Provide count and offset of queue range for each traffic class. In the format, **count@offset.** Queue ranges for each traffic classes cannot overlap and must be a contiguous range of queues.

hw Set to **1** to support hardware offload. Set to **0** to configure user specified values in software only.

mode Set to **channel** for full use of the mqprio options. Use **dcb** to offload only TC values and use hardware QOS defaults. Supported with 'hw' set to 1 only.

shaper Use **bw_rlimit** to set bandwidth rate limits for a traffic class. Use **dcb** for hardware QOS defaults. Supported with 'hw' set to 1 only.

min_rate
Minimum value of bandwidth rate limit for a traffic class.

max_rate
Maximum value of bandwidth rate limit for a traffic class.

AUTHORS

John Fastabend, <john.r.fastabend@intel.com>