NAME

ETF − Earliest TxTime First (ETF) Qdisc

SYNOPSIS

tc qdisc ... dev dev **parent** classid **[handle** major: **] etf clockid** clockid **[delta** delta_nsecs **] [deadline_mode] [offload]**

DESCRIPTION

The ETF (Earliest TxTime First) qdisc allows applications to control the instant when a packet should be dequeued from the traffic control layer into the netdevice. If **offload** is configured and supported by the network interface card, the it will also control when packets leave the network controller.

ETF achieves that by buffering packets until a configurable time before their transmission time (i.e. txtime, or deadline), which can be configured through the **delta** option.

The qdisc uses a rb-tree internally so packets are always 'ordered' by their txtime and will be dequeued following the (next) earliest txtime first.

It relies on the SO_TXTIME socket option and the SCM_TXTIME CMSG in each packet field to configure the behavior of time dependent sockets: the clockid to be used as a reference, if the expected mode of txtime for that socket is deadline or strict mode, and if packet drops should be reported on the socket's error queue. See **socket(7)** for more information.

The etf qdisc will drop any packets with a txtime in the past, or if a packet expires while waiting for being dequeued.

This queueing discipline is intended to be used by TSN (Time Sensitive Networking) applications, and it exposes a traffic shaping functionality that is commonly documented as "Launch Time" or "Time-Based Scheduling" by vendors and the documentation of network interface controllers.

ETF is meant to be installed under another qdisc that maps packet flows to traffic classes, one example is **mqprio(8).**

PARAMETERS

clockid

Specifies the clock to be used by qdisc's internal timer for measuring time and scheduling events. The qdisc expects that packets passing through it to be using this same **clockid** as the reference of their txtime timestamps. It will drop packets coming from sockets that do not comply with that.

For more information about time and clocks on Linux, please refer to **time(7)** and **clock_gettime(3).**

delta

After enqueueing or dequeueing a packet, the qdisc will schedule its next wake-up time for the next txtime minus this delta value. This means **delta** can be used as a fudge factor for the scheduler latency of a system. This value must be specified in nanoseconds. The default value is 0 nanoseconds.

deadline_mode

When **deadline_mode** is set, the qdisc will handle txtime with a different semantics, changed from a 'strict' transmission time to a deadline. In practice, this means during the dequeue flow **etf(8)** will set the txtime of the packet being dequeued to 'now'. The default is for this option to be disabled.

offload

When **offload** is set, **etf(8)** will try to configure the network interface so time-based transmission arbitration is enabled in the controller. This feature is commonly referred to as "Launch Time" or "Time-Based Scheduling" by the documentation of network interface controllers. The default is for this option to be disabled.

EXAMPLES

ETF is used to enforce a Quality of Service. It controls when each packets should be dequeued and transmitted, and can be used for limiting the data rate of a traffic class. To separate packets into traffic classes the user may choose **mqprio(8),** and configure it like this:

```
# tc qdisc add dev eth0 handle 100: parent root mqprio num_tc 3 \
      map 2 2 1 0 2 2 2 2 2 2 2 2 2 2 2 2 \
      queues 1@0 1@1 2@2 \
      hw 0
```

To replace the current queueing discipline by ETF in traffic class number 0, issue:

```
# tc qdisc replace dev eth0 parent 100:1 etf \
      clockid CLOCK_TAI delta 300000 offload
```

With the options above, etf will be configured to use CLOCK_TAI as its clockid_t, will schedule packets for 300 us before their txtime, and will enable the functionality on that in the network interface card. Deadline mode will not be configured for this mode.

AUTHORS

Jesus Sanchez-Palencia <jesus.sanchez-palencia@intel.com>
Vinicius Costa Gomes <vinicius.gomes@intel.com>