TC(8) Linux TC(8)

NAMEtc − show / manipulate traffic control settings SYNOPSIS tc qdisc [add | change | replace | link] dev DEV [parent qdisc-id | root] [handle qdisc-id] qdisc [qdisc specific parameters]
tc class [add | change | replace] dev DEV parent qdisc-id [classid class-id] qdisc [qdisc specific parameters]

tc filter [add | change | replace] dev DEV [parent qdisc-id | root] protocol protocol prio priority filtertype [filtertype specific parameters] flowid flow-id tc [-s | -d] qdisc show [dev DEV] tc [-s | -d] class show dev DEV tc filter show dev DEV DESCRIPTION Tc is used to configure Traffic Control in the Linux kernel. Traffic Control consists of the follow- ing:

SHAPINGWhen traffic is shaped, its rate of transmission is under control. Shaping may be more
than lowering the available bandwidth - it is also used to smooth out bursts in traffic for

better network behaviour. Shaping occurs on egress. SCHEDULING By scheduling the transmission of packets it is possible to improve interactivity for traffic

that needs it while still guaranteeing bandwidth to bulk transfers. Reordering is also

called prioritizing, and happens only on egress. POLICINGWhere shaping deals with transmission of traffic, policing pertains to traffic arriving.
Pol icing thus occurs on ingress. DROPPING Traffic exceeding a set bandwidth may also be dropped forthwith, both on ingress and on

egress. Processing of traffic is controlled by three kinds of objects: qdiscs, classes and filters. QDISCSqdisc is short for ’queueing discipline’ and it is elementary to understanding traffic control. Whenever the kernel needs to send a packet to an interface, it is enqueued to the qdisc config- ured for that interface. Immediately afterwards, the kernel tries to get as many packets as possible

from the qdisc, for giving them to the network adaptor driver.
A simple QDISC is the ’pfifo’ one, which does no processing at all and is a pure First In, First
Out queue. It does however store traffic when the network interface can’t handle it momentarily.

iproute2 16 December 2001 1

TC(8) Linux TC(8)

CLASSES Some qdiscs can contain classes, which contain further qdiscs - traffic may then be enqueued in

any of the inner qdiscs, which are within the classes. When the kernel tries to dequeue a packet from such a classful qdisc it can come from any of the classes. A qdisc may for example priori- tize certain kinds of traffic by trying to dequeue from certain classes before others. FILTERS A filter is used by a classful qdisc to determine in which class a packet will be enqueued. When- ever traffic arrives at a class with subclasses, it needs to be classified. Various methods may be

employed to do so, one of these are the filters. All filters attached to the class are called, until one
of them returns with a verdict. If no verdict was made, other criteria may be available. This dif-
fers per qdisc.

It is important to notice that filters reside within qdiscs - they are not masters of what happens. CLASSLESS QDISCS The classless qdiscs are: [p|b]fifo Simplest usable qdisc, pure First In, First Out behaviour. Limited in packets or in bytes. pfifo fastStandard qdisc for ’Advanced Router’ enabled kernels. Consists of a three-band queue

which honors Type of Service flags, as well as the priority that may be assigned to a

packet. red Random Early Detection simulates physical congestion by randomly dropping packets when nearing configured bandwidth allocation. Well suited to very large bandwidth appli-

cations. sfq Stochastic Fairness Queueing reorders queued traffic so each ’session’ gets to send a packet in turn. tbf The Token Bucket Filter is suited for slowing traffic down to a precisely configured rate. Scales well to large bandwidths. CONFIGURING CLASSLESS QDISCS In the absence of classful qdiscs, classless qdiscs can only be attached at the root of a device. Full

syntax:

tc qdisc add dev DEV root QDISC QDISC-PARAMETERS To remove, issue

tc qdisc del dev DEV root The pfifo fast qdisc is the automatic default in the absence of a configured qdisc. CLASSFUL QDISCS The classful qdiscs are:

CBQ Class Based Queueing implements a rich linksharing hierarchy of classes. It contains shaping elements as well as prioritizing capabilities. Shaping is performed using link idle

time calculations based on average packet size and underlying link bandwidth. The latter

may be ill-defined for some interfaces. HTB The Hierarchy Token Bucket implements a rich linksharing hierarchy of classes with an emphasis on conforming to existing practices. HTB facilitates guaranteeing bandwidth to

classes, while also allowing specification of upper limits to inter-class sharing. It contains iproute2 16 December 2001 2

TC(8) Linux TC(8) shaping elements, based on TBF and can prioritize classes. PRIO The PRIO qdisc is a non-shaping container for a configurable number of classes which are dequeued in order. This allows for easy prioritization of traffic, where lower classes are

only able to send if higher ones have no packets available. To facilitate configuration,

Type Of Service bits are honored by default. THEORY OF OPERATION Classes form a tree, where each class has a single parent. A class may have multiple children.

Some qdiscs allow for runtime addition of classes (CBQ, HTB) while others (PRIO) are created
with a static number of children.

Qdiscs which allow dynamic addition of classes can have zero or more subclasses to which traffic
may be enqueued.

Furthermore, each class contains a leaf qdisc which by default has pfifo behaviour though another qdisc can be attached in place. This qdisc may again contain classes, but each class can
have only one leaf qdisc.

When a packet enters a classful qdisc it can be classified to one of the classes within. Three crite- ria are available, although not all qdiscs will use all three: tc filtersIf tc filters are attached to a class, they are consulted first for relevant instructions. Filters

can match on all fields of a packet header, as well as on the firewall mark applied by

ipchains or iptables. See tc-filters (8). Type of Service Some qdiscs have built in rules for classifying packets based on the TOS field. skb->priority
Userspace programs can encode a class-id in the ’skb- >priority’ field using the SO PRI- ORITY option. Each node within the tree can have its own filters but higher level filters may also point directly

to lower classes.

If classification did not succeed, packets are enqueued to the leaf qdisc attached to that class.

Check qdisc specific manpages for details, however. NAMINGAll qdiscs, classes and filters have IDs, which can either be specified or be automatically assigned.

IDs consist of a major number and a minor number, separated by a colon.

QDISCSA qdisc, which potentially can have children, gets assigned a major number, called a ’han-

dle’, leaving the minor number namespace available for classes. The handle is expressed as

’10:’. It is customary to explicitly assign a handle to qdiscs expected to have children. CLASSESClasses residing under a qdisc share their qdisc major number, but each have a separate

minor number called a ’classid’ that has no relation to their parent classes, only to their
parent qdisc. The same naming custom as for qdiscs applies.

iproute2 16 December 2001 3

TC(8) Linux TC(8) FILTERSFilters have a three part ID, which is only needed when using a hashed filter hierarchy, for
which see tc-filters(8). UNITSAll parameters accept a floating point number, possibly followed by a unit.

Bandwidths or rates can be specified in:
kbps Kilobytes per second
mbps Megabytes per second
kbit Kilobits per second
mbit Megabits per second

bps or a bare number Bytes per second Amounts of data can be specified in:

kb or k Kilobytes mb or mMegabytes mbit Megabits

kbit Kilobits

b or a bare number Bytes. Lengths of time can be specified in:

s, sec or secs Whole seconds ms, msec or msecs Milliseconds us, usec, usecs or a bare number Microseconds. TC COMMANDS The following commands are available for qdiscs, classes and filter:

add Add a qdisc, class or filter to a node. For all entities, a parent must be passed, either by passing its ID or by attaching directly to the root of a device. When creating a qdisc or a

filter, it can be named with the handle parameter. A class is named with the classid parameter. remove A qdisc can be removed by specifying its handle, which may also be ’root’. All subclasses and their leaf qdiscs are automatically deleted, as well as any filters attached to them. change Some entities can be modified ’in place’. Shares the syntax of ’add’, with the exception that the handle cannot be changed and neither can the parent. In other words, change cannot move a node. replace Performs a nearly atomic remove/add on an existing node id. If the node does not exist yet it is created. iproute2 16 December 2001 4

TC(8) Linux TC(8) link Only available for qdiscs and performs a replace where the node must exist already.

HISTORY tc was written by Alexey N. Kuznetsov and added in Linux 2.2. SEE ALSO tc-cbq(8), tc-htb(8), tc-sfq(8), tc-red(8), tc-tbf(8), tc-pfifo(8), tc-bfifo(8), tc-pfifo fast(8), tc-filters(8) AUTHORManpage maintained by bert hubert (ahu@ds9a.nl)

iproute2 16 December 2001 5