NAME

tc − show / manipulate traffic control settings

SYNOPSIS

tc qdisc [**add** | **change** | **replace** | **link** | **delete**] **dev** DEV [**parent** qdisc-id | **root**] [**handle** qdisc-id] qdisc [qdisc specific parameters]

tc class [**add** | **change** | **replace** | **delete**] **dev** DEV **parent** qdisc-id [**classid** class-id] qdisc [qdisc specific parameters]

tc filter [**add** | **change** | **replace** | **delete**] **dev** DEV [**parent** qdisc-id | **root**] **protocol** protocol **prio** priority filtertype [filtertype specific parameters] **flowid** flow-id

tc [*FORMAT*] **qdisc show** [**dev** DEV]

tc [*FORMAT*] **class show dev** DEV

tc filter show dev DEV

tc [**-force**] **-b**[*atch*] [**filename**]

FORMAT := { −**s**[*tatistics*] | −**d**[*etails*] | −**r**[*aw*] | −**p**[*retty*] | −**i**[*ec*] }

DESCRIPTION

Tc is used to configure Traffic Control in the Linux kernel. Traffic Control consists of the following:

SHAPING
: When 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.

POLICING
: Whereas shaping deals with transmission of traffic, policing pertains to traffic arriving. Policing 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.

QDISCS

qdisc 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 configured 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.

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 prioritize 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. Whenever 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 differs 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_fast
: Standard 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 applications.

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 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, although 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 criteria are available, although not all qdiscs will use all three:

tc filters
: If 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.

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_PRIORITY 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.

NAMING

All 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. Both major and minor number are limited to 16 bits. There are two special values: root is signified by major and minor of all ones, and unspecified is all zeros.

QDISCS
: A qdisc, which potentially can have children, gets assigned a major number, called a 'handle', 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.

CLASSES
Classes 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.

FILTERS
Filters have a three part ID, which is only needed when using a hashed filter hierarchy.

PARAMETERS

The following parameters are widely used in TC. For other parameters, see the man pages for individual qdiscs.

RATES
Bandwidths or rates. These parameters accept a floating point number, possibly followed by a unit (both SI and IEC units supported).

bit or a bare number
: Bits per second

kbit Kilobits per second

mbit Megabits per second

gbit Gigabits per second

tbit Terabits per second

bps Bytes per second

kbps Kilobytes per second

mbps Megabytes per second

gbps Gigabytes per second

tbps Terabytes per second

To specify in IEC units, replace the SI prefix (k-, m-, g-, t-) with IEC prefix (ki-, mi-, gi- and ti-) respectively.

TC store rates as a 32-bit unsigned integer in bps internally, so we can specify a max rate of 4294967295 bps.

TIMES
Length of time. Can be specified as a floating point number followed by an optional unit:

s, sec or secs
: Whole seconds

ms, msec or msecs
: Milliseconds

us, usec, usecs or a bare number
: Microseconds.

TC defined its own time unit (equal to microsecond) and stores time values as 32-bit unsigned integer, thus we can specify a max time value of 4294967295 usecs.

SIZES Amounts of data. Can be specified as a floating point number followed by an optional unit:

b or a bare number
Bytes.

kbit Kilobits

kb or k
Kilobytes

mbit Megabits

mb or m
Megabytes

gbit Gigabits

gb or g
Gigabytes

TC stores sizes internally as 32-bit unsigned integer in byte, so we can specify a max size of 4294967295 bytes.

VALUES
Other values without a unit. These parameters are interpreted as decimal by default, but you can indicate TC to interpret them as octal and hexadecimal by adding a '0' or '0x' prefix respectively.

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.

delete A qdisc can be deleted 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.

link Only available for qdiscs and performs a replace where the node must exist already.

FORMAT

The show command has additional formatting options:

−s, **−stats**, **−statistics**
output more statistics about packet usage.

−d, **−details**
output more detailed information about rates and cell sizes.

−r, **−raw**
output raw hex values for handles.

−p, **−pretty**
decode filter offset and mask values to equivalent filter commands based on TCP/IP.

−iec print rates in IEC units (ie. 1K = 1024).

−b, **−b filename**, **−batch**, **−batch filename**
read commands from provided file or standard input and invoke them. First failure will cause termination of tc.

−force
don't terminate tc on errors in batch mode. If there were any errors during execution of the commands, the application return code will be non zero.

HISTORY

tc was written by Alexey N. Kuznetsov and added in Linux 2.2.

SEE ALSO

tc-bfifo(8), **tc-cbq**(8), **tc-choke**(8), **tc-codel**(8), **tc-drr**(8), **tc-ematch**(8), **tc-fq_codel**(8), **tc-hfsc**(7), **tc-hfsc**(8), **tc-htb**(8), **tc-pfifo**(8), **tc-pfifo_fast**(8), **tc-red**(8), **tc-sfb**(8), **tc-sfq**(8), **tc-stab**(8), **tc-tbf**(8),
User documentation at **http://lartc.org/**, but please direct bugreports and patches to: <**net-dev@vger.kernel.org**>

AUTHOR

Manpage maintained by bert hubert (ahu@ds9a.nl)