NAME

dcb − show / manipulate DCB (Data Center Bridging) settings

SYNOPSIS

dcb [*OPTIONS*] { **app** | **buffer** | **ets** | **maxrate** | **pfc** } { *COMMAND* | *help* }

dcb [**-force**] **-batch** *filename*

dcb [*OPTIONS*] **help**

OPTIONS

−n, **−-netns** <NETNS>
: switches **dcb** to the specified network namespace *NETNS*.

−V, **--Version**
: Print the version of the **dcb** utility and exit.

−b, **--batch** <FILENAME>
: Read commands from provided file or standard input and invoke them. First failure will cause termination of dcb.

−f, **--force**
: Don’t terminate dcb on errors in batch mode. If there were any errors during execution of the commands, the application return code will be non zero.

−i, **--iec**
: When showing rates, use ISO/IEC 1024-based prefixes (Ki, Mi, Bi) instead of the 1000-based ones (K, M, B).

−j, **--json**
: Generate JSON output.

−N, **--Numeric**
: If the subtool in question translates numbers to symbolic names in some way, suppress this translation.

−p, **--pretty**
: When combined with -j generate a pretty JSON output.

−s, **--statistics**
: If the object in question contains any statistical counters, shown them as part of the "show" output.

OBJECTS

app - Configuration of application priority table

buffer - Configuration of port buffers

ets - Configuration of ETS (Enhanced Transmission Selection)

maxrate
- Configuration of per-TC maximum transmit rate

pfc - Configuration of PFC (Priority-based Flow Control)

COMMANDS

A *COMMAND* specifies the action to perform on the object. The set of possible actions depends on the object type. As a rule, it is possible to **show** objects and to invoke topical **help,** which prints a list of available commands and argument syntax conventions.

ARRAY PARAMETERS

Like commands, specification of parameters is in the domain of individual objects (and their commands) as well. However, much of the DCB interface revolves around arrays of fixed size that specify one value per some key, such as per traffic class or per priority. There is therefore a single syntax for adjusting elements of these arrays. It consists of a series of *KEY***:***VALUE* pairs, where the meaning of the individual keys and values depends on the parameter.

The elements are evaluated in order from left to right, and the latter ones override the earlier ones. The elements that are not specified on the command line are queried from the kernel and their current value is retained.

As an example, take a made-up parameter tc-juju, which can be set to charm traffic in a given TC with either good luck or bad luck. *KEY* can therefore be 0..7 (as is usual for TC numbers in DCB), and *VALUE* either of **none**, **good**, and **bad**. An example of changing a juju value of TCs 0 and 7, while leaving all other intact, would then be:

dcb foo set dev eth0 tc-juju 0:good 7:bad

A special key, **all**, is recognized which sets the same value to all array elements. This can be combined with the usual single-element syntax. E.g. in the following, the juju of all keys is set to **none**, except 0 and 7, which have other values:

dcb foo set dev eth0 tc-juju all:none 0:good 7:bad

EXIT STATUS

Exit status is 0 if command was successful or a positive integer upon failure.

SEE ALSO

dcb-app(8), **dcb-buffer**(8), **dcb-ets**(8), **dcb-maxrate**(8), **dcb-pfc**(8)

REPORTING BUGS

Report any bugs to the Network Developers mailing list **<netdev@vger.kernel.org>** where the development and maintenance is primarily done. You do not have to be subscribed to the list to send a message there.

AUTHOR

Petr Machata <me@pmachata.org>