# Egress and Ingress Example

We need some iptables lib to do this test. (libxt\_CLASSIFY.so libxt\_dscp.so libxt\_tos.so)

The lib is not contained in current software. If you want to test, we can send you these lib and you should put them into /usr/lib/iptables. These libs will be contained in the future release.

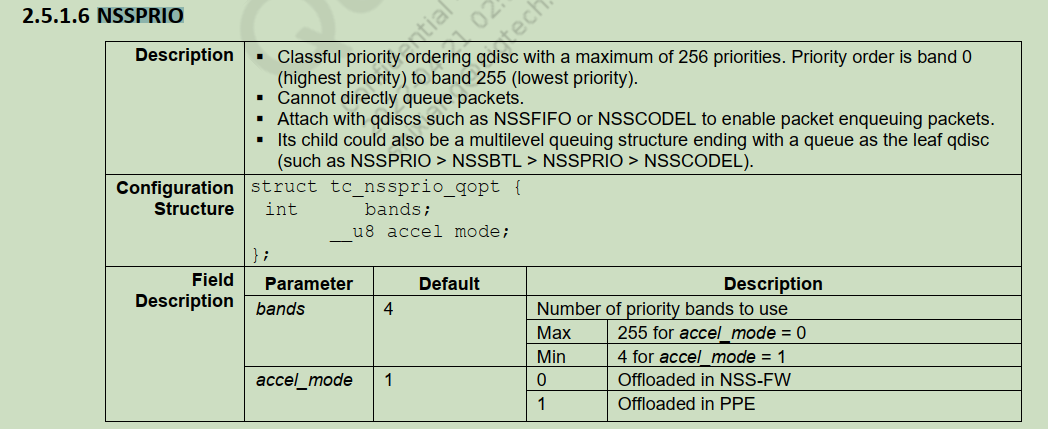
The first test, egress test.

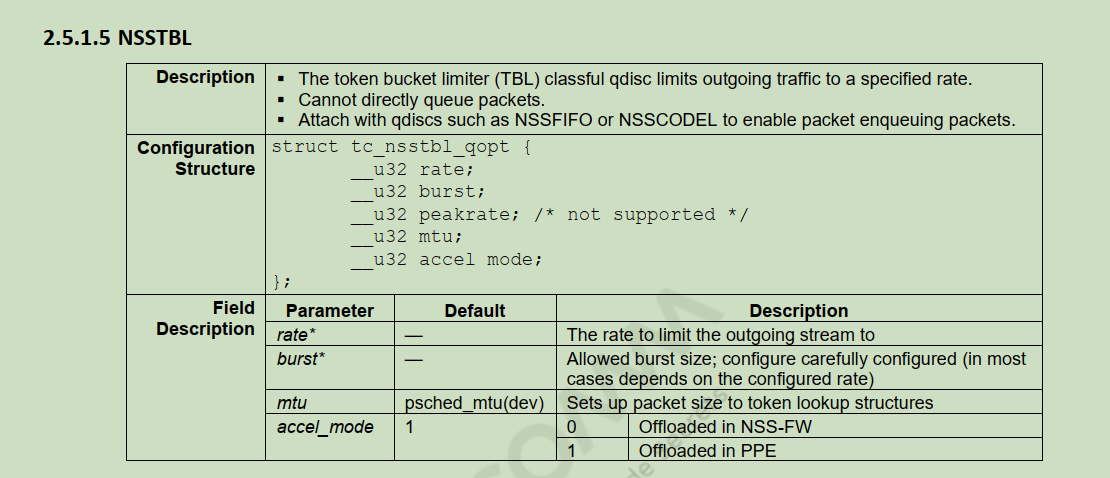
|  |
| --- |
| egress qos test  iptables -t mangle -I OUTPUT -m tos --tos 0x0 -o eth0 -j CLASSIFY --set-class 100:0  tc qdisc add dev eth0 root handle 1: nssprio bands 2  tc qdisc add dev eth0 parent 1:1 handle 10: nsstbl rate 10Mbit burst 30K  tc qdisc add dev eth0 parent 1:2 handle 20: nsstbl rate 1000Mbit burst 30K  tc qdisc add dev eth0 parent 10: handle 100: nsspfifo limit 100  tc qdisc add dev eth0 parent 20: handle 200: nsspfifo limit 100 set\_default |

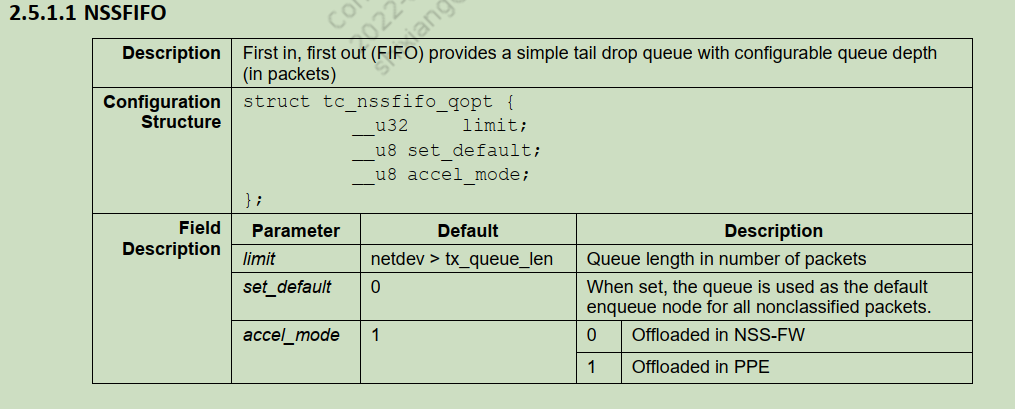
The iptables will check the packet tos. if it is 0, it will set the mark 100:0 on that packet.

These tc commands set up 2 queues, 100: and 200:, with 200: as the default queue where all  
nonclassified packets enqueue. In this configuration, the 100: queue attaches at higher  
priority compared to queue 200:. It ensures that traffic directed to queue 100: (that is, flows  
with tos=0) are scheduled with higher priority compared to other flows in the system. Use nssprio to create two queues and use nsstbl to set the speed limitation of the queue.

From this example, the packet that tos is 0 will go to 100:0 that speed is set to 10Mbit







The second test, ingress test.

|  |
| --- |
| tc qdisc add dev eth0 handle ffff: ingress  /etc/init.d/qca-nss-mirred start  tc filter add dev eth0 parent ffff: u32 match u32 0 0 action nssmirred redirect dev ifb1 fromdev eth0  tc qdisc add dev ifb1 root handle 1: nssprio bands 3  tc qdisc add dev ifb1 parent 1:1 handle 11: nsspfifo limit 100  tc qdisc add dev ifb1 parent 1:2 handle 12: nsspfifo limit 100  tc qdisc add dev ifb1 parent 1:3 handle 13: nsspfifo limit 100 set\_default  tc filter add dev ifb1 parent 1: protocol ip u32 match ip dst 192.168.1.90/4 flowid 11:0  ifconfig ifb1 up |

The Linux traffic control utility (**tc**) supports ingress Qdisc. For traffic received on an interface, this ingress Qdisc is traversed. Ingress Qdisc supports only policing through **tc** filters and shapers cannot be attached to it. Ingress Qdisc is used together with Intermediate Functional Block interface for attaching shapers and shaping the ingress traffic.

**The first command**: Attach the ingress Qdisc to the given interface where ingress shaping is needed

**The second command**: It will create the virtual interface, ifb1.

**The third command**: Attach the filter rule to redirect ingress traffic to an IFB interface. Use the custom action object **nssmirred** to redirect traffic. This action triggers the virtual interface creation and redirection through **nssmirred.ko** in NSS.

The command then creates three queues. The 13: is the default queue for packet. 12: and 11: are the high priority queues. The last command sets the ip 192.168.1.90/4 to go to the high priority queue 11:0.

Some command to check the tc status.

|  |  |
| --- | --- |
| Method | example |
| check the interface rules | tc -s qdisc show dev eth0 |
| delete the rules on interface | tc qdisc del dev eth0 root |
| change the rules when the tc is running | tc qdisc change dev eth0 root handle 1: nsstbl rate 10Mbit burst 30K |