Chapter 4

The DiffServ MIB and the Linux Kernel

4.1 Representing information from the kernel

The prototyping in this assignment is focused on the monitoring part of the manager-agent paradigm, i.e. the manager retrieves some values from the agent. Therefore the implementation of the agent needs to gather information when it is requested to do so (or on its own). In the case of the DiffServ MIB agent, this means that the agent has to get information from the kernel using the rtnetlink messages. This section covers the various tables from the DiffServ MIB and explains what needs to be done in order to retrieve the requested information.

4.1.1 Data Path Table

This table merely is a starting point for the DiffServ management information. The network interfaces are numbered according to the if Table numbering, which happens to be the same as the internal numbering used in the Linux kernel. This information is retrieved using a RTM-GETLINK message. No conversion needs to be done at all, though parsing of the resulting message remains necessary of course.

4.1.2 Classifier tables

In the MIB there is the SixTuple Classifier which makes it possible to represent any part of the IP and transport layer headers in the MIB, like IP addresses, DSCPs and port numbers. The Classifier tables are indexed by two separate identifiers, enumerating the classifier entries and the classifier element entries. In Linux, there is no unique identification for a filter. Elements within a single filter are identified with a handle though. Multiple filters belonging to the same queueing discipline or class are ordered by a numerical priority value. Thus a DiffServ MIB implementation must take care of assigning unique identifiers to the filters and their elements itself.

4.1.3 Meter tables

The Meter tables can be found in the diffServMeter and diffServTB-Param subtrees in the MIB. Linux uses only three primitives when dealing with traffic
control: qdisc, filter and class. None of them is exactly the same as a Meter in the DiffServ architecture. However, the Classifier element in that picture, which is not a real element as it not formally defined in the TC architecture, is what the filter primitive can do in terms of functionality. A conclusion and solution to this problem can easily be drawn: the RTM-FILTER messages can be used to gain knowledge about policing in the kernel as well. This is something that typically only occurs at edge routers at the boundary of a diffserv domain; core routers use filters only for determining the DSCP value.

![Figure 4.1: Classsifying,policing and markind Diffserv domain](image)

### 4.1.4 Action space

There are basically two operations that belong to these tables: marking with some DSCP value, and counting the packets. Marking operation in the DiffSer architecture is part of the Class element in the TC architecture. The corresponding primitive, class, provides an implementor with the necessary operations, using the RTM-TCLASS messages. Counting is achieved in the TC architecture using a packet counting FIFO that can be used as an inner queueing discipline. This is controlled with the qdisc primitive. In the Linux kernel this special FIFO queue is enhanced with a limit on its size, but the DiffServ MIB does not support this for the Counting operation. Indexing these tables with the right values is once again up to the agent, though the handles and classid’s that are used internally by the Linux TC engine might be helpful.

### 4.1.5 queing tables

In the MIB queueing falls apart in three subcategories: algorithmic dropper management, and queuing and scheduling management. They are all part...
of both the class and qdisc primitives in the Linux TC architecture. Class-Based Queuing (CBQ) is often used in Linux to get this functionality.

The TC implementation provides a so-called Weighted Round Robin (WRR) scheduling method as part of CBQ, and parameters to this algorithm are accessed using the RTM-QDISC messages. WRR can be represented in the MIB in the Scheduler table, using an diffServAssuredRateEntry to store the parameters. The diffServShapingRate table is not used, as there is no corresponding implementation in the Linux Network Traffic Control engine. Administration like indexing values for these tables is up to the DiffServ MIB agent. Theoretically it is possible to share queues in TC, but this is very complicated and introduces other problems.
References


http://en.wikipedia.org/wiki/Management_information_base