# The Meter MIB

**Nevil Brownlee**
**Computer Centre**
**The University of Auckland**

***Toronto IETF, Thu 28 Jul 94***

# *Introduction*

- ***Accounting Meter***
  - **Small host, attached to a network**
  - **Observes data packets**
  - **Builds a table of traffic flows**
- ***Traffic Flows***
  - **A *flow* is a stream of packets between two network hosts**
  - **Flows are bi-directional, *Source* and *Dest* are specified by the user**
  - **Each 'end' is described by a set of attributes**
  - **Flow has other attributes, e.g. octet and PDU counts, creation and last-use times**

# *MIB Outline*

- **Current version documented in I-D:**
  **draft-brownlee-acct-meter-mib-00.txt**
- **Currently experimental.99 in the MIB tree**
- **Four sections**
  - **acctControl** — **Meter control**
  - **acctFlowdata** — **Traffic flows**
  - **acctRuledata** } **Packet-matching rules**
  - **acctActiondata** } **and actions**

# *Definitions*

- ***Address Attributes***
  - ***Adjacent* Address: link layer**
  - ***Peer* Address: network layer**
    **(IP, IPX, EtherTalk, DECnet, CLNS)**
  - ***Transport* Address: varies with protocol (IP port number)**
- ***Address Masks***
  - **Each address attribute also has a mask**
  - ***Mask* bits indicate which address bits are significant**

# *Rule Table (i)*

- ***Rules*** **describe the flows of interest as a set of *patterns***
- **Meter matches these patterns against every packet it processes**
- **Each address attribute also has a *Mask***
- **Mask bits indicate which address bits are significant**
- **Packet processing –**
  - ***Classify*****: match patterns to determine *flow.***
    **No match –> ignore this packet**
  - ***Search*****: make sure flow is in the meter's tables**
  - ***Count*****: increment flow counters, set times**

# *Rule Table (ii)*

- **A rule is a row in the meter's rule table**
- **Each rule has the form**
  **{ Attribute, Mask, Value, Action, Index }**
- ***Attribute*** **is ANDed with** ***Mask*** **and compared to** ***Value*****. Equal –> execute** ***Action*****.**
  ***Index*** **is a parameter for Action**
- **Action is performed when a rule is matched**
  - ***Count / Tally / Aggregate***
  - ***Matching flow control***
  - ***Matching subroutines***

# *Actions*

- ***Succeed***: Stop matching, ignore the packet
- ***PushTo***: Index = next rule to test.
  (Remember matched rules)
- ***Count:*** }
- ***Tally***: } Search, Insert, Update flow
- ***Aggregate***: }

# *Count / Tally / Aggregate*

- **Each action is a prototype flow; masks and values may be specified for any attribute**
- ***Aggregate* –> Use masks and values from Action**
- ***Tally* –> Process packet as one of a set of Tally flows. Masks specified by Action, Values from the packet. e.g. test PeerAddress = Class B, tally with Class C mask**
- ***Count* –> Process this packet as one of a set of *Count* flows. Masks and values from matched rules**

# *Matching Subroutines*

- **Aim is to allow more powerful patterns, i.e. use less rules to describe more flows**
- ***Fail:*** **Causes match to fail. This allows meter to swap the packet's Source and Dest then retry the match**
- ***Assign:*** **}**
- ***Gosub:*** **} Allow rule-matching subroutines**
- ***Return n:*** **}**
- **Shorter rule tables provide faster pattern matching!**

# *Flow Table*

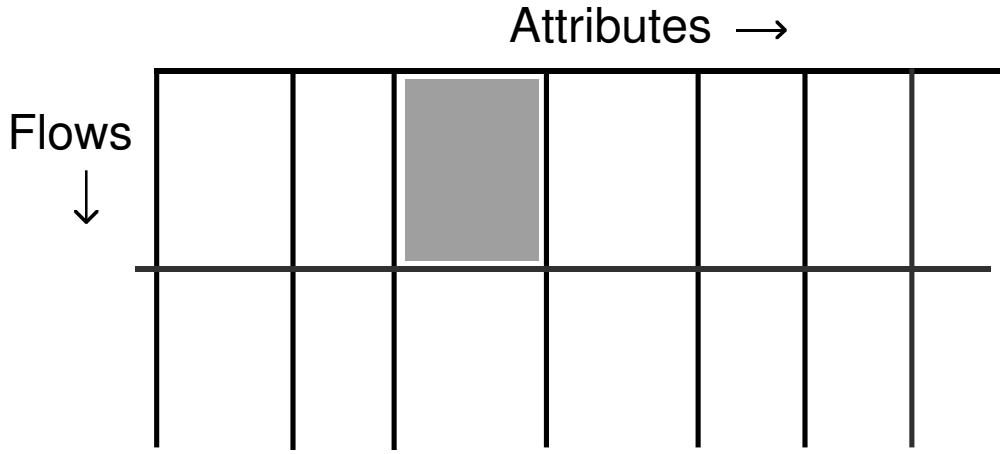
- **Flows stored as rows in a matrix**
  - **Rows may appear and disappear**
- ***Creation* Table**
  - **Window into the flow table, indexed by Create Time, i.e. time the flow's first packet was seen**
- ***Activity* Table**
  - **Window into the flow table, indexed by Create Time, i.e. time the flow's first packet was seen**

# *Creation and Activity Tables*

- **Allow collection of flow data by several collectors at different intervals**
- **SNMP object IDs for ‘creation’ and ‘activity’ table entries include an index and a time**
- **Meter returns the first SNMP object with both time and index greater than the requested OID values**
- **This allows a collector to find all the flows created or active after a specified time**

# *Bulk Data Collection*

- **Flow Table is a 2-D array: Attributes x Flows**
- **Meter uses an ASN.1 opaque object ('column blob') to collect part of an attribute column**




- **Collector can read specified set of attributes for flows active after a given time – this minimises SNMP traffic**

# *Meter Control*

- **Meter maintains table of collector IP addresses and collection times – acctCollectorInfo**
- **Flows which haven't been active since last recorded collection can be recovered**
- ***InactivityTimeout* variable specifies minimum lifetime for flows**
- ***HighWaterMark* variable warns meter garbage collection should be speeded up. NeTraMet does this by reducing InactivityTimeout**

# *Rule Set Info*

- **acctRuleSetInfo holds sizes of rule and action tables**
- **Setting these variables allocates space for the tables**
- **Set 1 is the meter's default, which the meter runs when sarted up. It can't be altered**
- **acctCurrentRuleSet specifies set meter is using. Set this to change the rule set**

# *Other Control Variables*

- **Emergency Rule Set. Intended to allow the meter to switch to a 'coarser' rule table after a declareDataLoss trap**
- **Sampling Rate. Allows user to specify '1 in every n' packets to be processed. Default is n=1, i.e. every packet is processed**

# *Questions*

- **Should there be a variable to specify the minimum number of collections required before a flow can be recovered?**
- **Should the meter remember its rule sets and restart with them after a power failure?**
- **Should there be a trap to let the meter say ‘restarting after a power outage?’**