Network Working Group   S. Handelman
Request for Comments: 2724   S. Stibler
Category: Experimental   IBM
                                                          N. Brownlee
                                            The University of Auckland
                                                             G. Ruth
                                                GTE Internetworking
                                                        October 1999

            RTFM: New Attributes for Traffic Flow Measurement

Abstract

   The RTFM Traffic Measurement Architecture provides a general
   framework for describing and measuring network traffic flows. Flows
   are defined in terms of their Address Attribute values and measured
   by a 'Traffic Meter'. This document discusses RTFM flows and the
   attributes which they can have, so as to provide a logical framework
   for extending the architecture by adding new attributes.

   Extensions described include Address Attributes such as DSCodePoint,
   SourceASN and DestASN, and Group Attributes such as short-term bit
   rates and turnaround times. Quality of Service parameters for
   Integrated Services are also discussed.

Table of Contents

[1](#) Introduction

The Real-Time Flow Measurement (RTFM) Working Group (WG) has
developed a system for measuring and reporting information about
traffic flows in the Internet. This document explores the definition
of extensions to the flow measurements as currently defined in
[[RTFM-ARC](#)]. The new attributes described in this document will be
useful for monitoring network performance and will expand the scope
of RTFM beyond simple measurement of traffic volumes. A companion
document to this memo will be written to define MIB structures for
the new attributes.

This memo was started in 1996 to advance the work of the RTFM group.
The goal of this work is to produce a simple set of abstractions,
which can be easily implemented and at the same time enhance the
value of RTFM Meters. This document also defines a method for
organizing the flow abstractions to augment the existing RTFM flow
table.

Implementations of the RTFM Meter have been done by Nevil Brownlee in
the University of Auckland, NZ, and Stephen Stibler and Sig Handelman
at IBM in Hawthorne, NY, USA. The RTFM WG has also defined the role
of the Meter Reader whose role is to retrieve flow data from the
Meter.

Note on flows and positioning of meters:

   A flow as it traverses the Internet may have some of its
   characteristics altered as it travels through Routers, Switches,
   and other network units. It is important to note the spatial
   location of the Meter when referring to attributes of a flow. An
   example, a server may send a sequence of packets with a definite
   order, and inter packet timing with a leaky bucket algorithm. A
   meter reading downstream of the leaky bucket would record a set
   with minimal inter packet timing due to the leaky bucket. At the
   client's location, the packets may arrive out of sequence, with

     the timings altered. A meter at the client's location would
     record different attributes for the same flow.

1.1 RTFM's Definition of Flows

   The RTFM Meter architecture views a flow as a set of packets between
   two endpoints (as defined by their source and destination attribute
   values and start and end times), and as BI-DIRECTIONAL (i.e. the
   meter effectively monitors two sub-flows, one in each direction).

   Reasons why RTFM flows are bi-directional:

     - The WG is interested in understanding the behavior of sessions
        between endpoints.

     - The endpoint attribute values (the "Address" and "Type" ones)
        are the same for both directions; storing them in bi-
        directional flows reduces the meter's memory demands.

     - 'One-way' (uni-directional) flows are a degenerate case.
        Existing RTFM meters can handle this by using one of the
        computed attributes (e.g. FlowKind) to indicate direction.

1.2 RTFM's Current Definition of Flows and their Attributes

   Flows, as described in the "Architecture" document [RTFM-ARC] have
   the following properties:

   a. They occur between two endpoints, specified as sets of attribute
      values in the meter's current rule set. A flow is completely
      identified by its set of endpoint attribute values.

   b. Each flow may also have values for "computed" attributes (Class
      and Kind). These are directly derived from the endpoint attribute
      values.

   c. A new flow is created when a packet is to be counted that does not
      match the attributes of an existing flow. The meter records the
      time when this new flow is created.

   d. Attribute values in (a), (b) and (c) are set when the meter sees
      the first packet for the flow, and are never changed.

   e. Each flow has a "LastTime" attribute, which indicates the time the
      meter last saw a packet for the flow.

f. Each flow has two packet and two byte counters, one for each flow
   direction (Forward and Backward). These are updated as packets
   for the flow are observed by the meter.

g. ALL the attributes have (more or less) the same meaning for a
   variety of protocols; IPX, AppleTalk, DECnet and CLNS as well as
   TCP/IP.

Current flow attributes - as described above - fit very well into the
SNMP data model. They are either static, or are continuously updated
counters. They are NEVER reset. In this document they will be
referred to as "old-style" attributes.

It is easy to add further "old-style" attributes, since they don't
require any new features in the architecture. For example:

- Count of the number of "lost" packets (determined by watching
  sequence number fields for packets in each direction; only
  available for protocols which have such sequence numbers).

- In the future, RTFM could coordinate directly with the Flow
  Label from the IPv6 header.

1.3 RTFM Flows, Integrated Services, IPPM and Research in Flows

The concept of flows has been studied in various different contexts.
For the purpose of extending RTFM, a starting point is the work of
the Integrated Services WG. We will measure quantities that are often
set by Integrated Services configuration programs. We will look at
the work of the Benchmarking/IP Performance Metrics Working Group,
and also look at the work of Claffy, Braun and Polyzos [C-B-P]. We
will demonstrate how RTFM can compute throughput, packet loss, and
delays from flows.

An example of the use of capacity and performance information is
found in "The Use of RSVP with IETF Integrated Services" [IIS-RSVP].
RSVP's use of Integrated Services revolves around Token Bucket Rate,
Token Bucket Size, Peak Data Rate, Minimum Policed Unit, Maximum
Packet Size, and the Slack term. These are set by TSpec, ADspec and
FLowspec (Integrated Services Keywords), and are used in
configuration and operation of Integrated Services. RTFM could
monitor explicitly Peak Data Rate, Minimum Policed Unit, Maximum
Packet Size, and the Slack term. RTFM could infer details of the
Token Bucket. The WG will develop measures to work with these
service metrics. An initial implementation of IIS Monitoring has
been developd at CEFRIEL in Italy [IIS-ACCT].

RTFM will work with several traffic measurements identified by IPPM
[IPPM-FRM]. There are three broad areas in which RTFM is useful for
IPPM.

   - An RTFM Meter could act as a passive device, gathering traffic
      and performance statistics at appropriate places in networks
      (server or client locations).

   - RTFM could give detailed analyses of IPPM test flows that pass
      through the Network segment that RTFM is monitoring.

   - RTFM could be used to identify the most-used paths in a network
      mesh, so that detailed IPPM work could be applied to these most
      used paths.

2 Flow Abstractions

   Performance attributes include throughput, packet loss, delays,
   jitter, and congestion measures. RTFM will calculate these
   attributes in the form of extensions to the RTFM flow attributes
   according to three general classes:

   - 'Trace', attributes of individual packets in a flow or a
      segment of a flow (e.g. last packet size, last packet arrival
      time).

   - 'Aggregate', attributes derived from the flow taken as a whole
      (e.g. mean rate, max packet size, packet size distribution).

   - 'Group', attributes that depend on groups of packet values
      within the flow (e.g. inter-arrival times, short-term traffic
      rates).

   Note that attributes within each of these classes may have various
   types of values - numbers, distributions, time series, and so on.

2.1 Meter Readers and Meters

   A note on the relation between Meter Readers and Meters.

   Several of the measurements enumerated below can be implemented by a
   Meter Reader that is tied to a meter with very short response time
   and very high bandwidth. If the Meter Reader and Meter can be
   arranged in such a way, RTFM could collect Packet Traces with time
   stamps and provide them directly to the Meter Reader for further
   processing.

   A more useful alternative is to have the Meter calculate some flow
   statistics locally. This allows a looser coupling between the Meter
   and Meter Reader. RTFM will monitor an 'extended attribute'
   depending upon settings in its Rule table. RTFM will not create any
   "extended attribute" data without explicit instructions in the Rule
   table.

2.2 Attribute Types

   Section 2 described three different classes of attributes; this
   section considers the "data types" of these attributes.

   Packet Traces (as described below) are a special case in that they
   are tables with each row containing a sequence of values, each of
   varying type. They are essentially 'compound objects' i.e. lists of
   attribute values for a string of packets.

   Aggregate attributes are like the 'old-style' attributes. Their
   types are:

      - Addresses, represented as byte strings (1 to 20 bytes long)

      - Counters, represented as 64-bit unsigned integers

      - Times, represented as 32-bit unsigned integers

   Addresses are saved when the first packet of a flow is observed.
   They do not change with time, and they are used as a key to find the
   flow's entry in the meter's flow table.

   Counters are incremented for each packet, and are never reset. An
   analysis application can compute differences between readings of the
   counters, so as to determine rates for these attributes. For
   example, if we read flow data at five-minute intervals, we can
   calculate five-minute packet and byte rates for the flow's two
   directions.

   Times are derived from the FirstTime for a flow, which is set when
   its first packet is observed. LastTime is updated as each packet in
   the flow is observed.

   All the above types have the common feature that they are expressed
   as single values. At least some of the new attributes will require
   multiple values. If, for example, we are interested in inter-packet
   time intervals, we can compute an interval for every packet after the
   first. If we are interested in packet sizes, a new value is obtained
   as each packet arrives. When it comes to storing this data we have
   two options:

      - As a distribution, i.e. in an array of 'buckets'. This method
         is a compact representation of the data, with the values being
         stored as counters between a minimum and maximum, with defined
         steps in each bucket. This fits the RTFM goal of compact data
         storage.

      - As a sequence of single values. This saves all the
         information, but does not fit well with the RTFM goal of doing
         as much data reduction as possible within the meter.

   Studies which would be limited by the use of distributions might well
   use packet traces instead.

   A method for specifying the distribution parameters, and for encoding
   the distribution so that it can be easily read, is described in
   section 3.2.

2.3 Packet Traces

   The simplest way of collecting a trace in the meter would be to have
   a new attribute called, say, "PacketTrace". This could be a table,
   with a column for each property of interest. For example, one could
   trace:

      - Packet Arrival time (TimeTicks from sysUpTime, or microseconds
         from FirstTime for the flow).

      - Packet Direction (Forward or Backward)

      - Packet Sequence number (for protocols with sequence numbers)

      - Packet Flags (for TCP at least)

   Note: The following implementation proposal is for the user who is
   familiar with the writing of rule sets for the RTFM Meter.

      To add a row to the table, we only need a rule which PushPkts the
      PacketTrace attribute. To use this, one would write a rule set
      which selected out a small number of flows of interest, with a
      'PushPkt PacketTrace' rule for each of them. A MaxTraceRows
      default value of 2000 would be enough to allow a Meter Reader to
      read one-second ping traces every 10 minutes or so. More
      realistically, a MaxTraceRows of 500 would be enough for one-
      minute pings, read once each hour.

   Packet traces are already implemented by the RMON MIB [RMON-MIB,
   RMON2-MIB], in the Packet Capture Group. They are therefore a low
   priority for RTFM.

2.4 Aggregate Attributes

   RTFM's "old-style" flow attributes count the bytes and packets for
   packets which match the rule set for an individual flow. In addition
   to these totals, for example, RTFM could calculate Packet Size
   statistics. This data can be stored as distributions, though it may
   sometimes be sufficient to simply keep a maximum value.

   As an example, consider Packet Size. RTFM's packet flows can be
   examined to determine the maximum packet size found in a flow. This
   will give the Network Operator an indication of the MTU being used in
   a flow. It will also give an indication of the sensitivity to loss
   of a flow, for losing large packets causes more data to be
   retransmitted.

   Note that aggregate attributes are a simple extension of the 'old-
   style' attributes; their values are never reset. For example, an
   array of counters could hold a 'packet size' distribution. The
   counters continue to increase, a meter reader will collect their
   values at regular intervals, and an analysis application will compute
   and display distributions of the packet size for each collection
   interval.

2.5 Group Attributes

   The notion of group attributes is to keep simple statistics for
   measures that involve more than one packet. This section describes
   some group attributes which it is feasible to implement in a traffic
   meter, and which seem interesting and useful.

   Short-term bit rate - The data could also be recorded as the maximum
   and minimum data rate of the flow, found over specific time periods
   during the lifetime of a flow; this is a special kind of
   'distribution'. Bit rate could be used to define the throughput of a
   flow, and if the RTFM flow is defined to be the sum of all traffic in
   a network, one can find the throughput of the network.

   If we are interested in '10-second' forward data rates, the meter
   might compute this for each flow of interest as follows:

     - maintain an array of counters to hold the flow's 10-second data
       rate distribution.

     - every 10 seconds, compute and save 10-second octet count, and
       save a copy of the flow's forward octet counter.

To achieve this, the meter will have to keep a list of aggregate
flows and the intervals at which they require processing. Careful
programming is needed to achieve this, but provided the meter is not
asked to do it for very large numbers of flows, it has been
successfully implemented.

Inter-arrival times. The Meter knows the time that it encounters
each individual packet. Statistics can be kept to record the inter-
arrival times of the packets, which would give an indication of the
jitter found in the Flow.

Turn-around statistics. Sine the Meter knows the time that it
encounters each individual packet, it can produce statistics of the
time intervals between packets in opposite directions are observed on
the network. For protocols such as SNMP (where every packet elicits
an answering packet) this gives a good indication of turn-around
times.

Subflow analysis. Since the choice of flow endpoints is controlled
by the meter's rule set, it is easy to define an aggregate flow, e.g.
"all the TCP streams between hosts A and B." Preliminary
implementation work suggests that - at least for this case - it
should be possible for the meter to maintain a table of information
about all the active streams. This could be used to produce at least
the following attributes:

   - Number of streams, e.g. streams active for n-second intervals.
     Determined for TCP and UDP using source-dest port number pairs.

   - Number of TCP bytes, determined by taking difference of TCP
     sequence numbers for each direction of the aggreagate flow.

IIS attributes. Work at CEFRIEL [IIS-ACCT] has produced a traffic
meter with a rule set modified 'on the fly' so as to maintain a list
of RSVP-reserved flows. For such flows the following attributes have
been implemented (these quantities are defined in [GUAR-QOS]):

     - QoSService: Service class for the flow
                              (guaranteed, controlled load)
     - QoSStyle: Reservation setup style
                              (wildcard filter, fixed filter,
                              shared explicit)
     - QoSRate: [byte/s] rate for flows with
                              guaranteed service
     - QoSSlackTerm: [microseconds] Slack Term QoS parameter
                              for flows with guaranteed service
     - QoSTokenBucketRate: [byte/s] Token Bucket Rate QoS parameter
                              for flows with guaranteed or
                              controlled load service

     The following are also being considered:

     - QoSTokenBucketSize: [byte] Size of Token Bucket

     - QoSPeakDataRate: [byte/s] Maximum rate for incoming data

     - QoSMinPolicedUnit: [byte] IP datagrams less than this are
                              counted as being this size
     - QoSMaxDatagramSize: [byte] Size of biggest datagram which
                              conforms to the traffic specification

2.6 Actions on Exceptions

   Some users of RTFM have requested the ability to mark flows as having
   High Watermarks. The existence of abnormal service conditions, such
   as non-ending flow, a flow that exceeds a given limit in traffic
   (e.g. a flow that is exhausting the capacity of the line that carries
   it) would cause an ALERT to be sent to the Meter Reader for
   forwarding to the Manager. Operations Support could define service
   situations in many different environments. This is an area for
   further discussion on Alert and Trap handling.

3 Extensions to the 'Basic' RTFM Meter

   The Working Group has agreed that the basic RTFM Meter will not be
   altered by the addition of the new attributes of this document. This
   section describes the extensions needed to implement the new
   attributes.

3.1 Flow table extensions

   The architecture of RTFM has defined the structure of flows, and this
   memo does not change that structure. The flow table could have
   ancillary tables called "Distribution Tables" and "Trace Tables,"

   these would contain rows of values and or actions as defined above.
   Each entry in these tables would be marked with the number of its
   corresponding flow in the RTFM flow table.

   Note: The following section is for the user who is familiar with the
   writing of rule sets for the RTFM Meter.

      In order to identify the data in a Packet Flow Table, the
      attribute name could be pushed into a string at the head of each
      row. For example, if a table entry has "To Bit Rate" for a
      particular flow, the "ToBitRate" string would be found at the head
      of the row. (An alternative method would be to code an
      identification value for each extended attribute and push that
      value into the head of the row.) See section 4. for an inital
      set of ten extended flow attributes.

3.2 Specifying Distributions in RuleSets

   At first sight it would seem neccessary to add extra features to the
   RTFM Meter architecture to support distributions. This, however, is
   not neccessarily the case.

   What is actually needed is a way to specify, in a ruleset, the
   distribution parameters. These include the number of counters, the
   lower and upper bounds of the distribution, whether it is linear or
   logarithmic, and any other details (e.g. the time interval for
   short-term rate attributes).

   Any attribute which is distribution-valued needs to be allocated a
   RuleAttributeNumber value. These will be chosen so as to extend the
   list already in the RTFM Meter MIB document [RTFM-MIB].

   Since distribution attributes are multi-valued it does not make sense
   to test them. This means that a PushPkt (or PushPkttoAct) action
   must be executed to add a new value to the distribution. The old-
   style attributes use the 'mask' field to specify which bits of the
   value are required, but again, this is not the case for
   distributions. Lastly, the MatchedValue ('value') field of a PushPkt
   rule is never used. Overall, therefore, the 'mask' and 'value'
   fields in the PushPkt rule are available to specify distribution
   parameters.

   Both these fields are at least six bytes long, the size of a MAC
   address. All we have to do is specify how these bytes should be
   used! As a starting point, the following is proposed (bytes are
   numbered left-to-right.

Mask bytes:
```
     1     Transform          1 = linear, 2 = logarithmic
     2     Scale Factor       Power of 10 multiplier for Limits
                                 and Counts
   3-4     Lower Limit        Highest value for first bucket
   5-6     Upper Limit        Highest value for last bucket
```

Value bytes:
```
     1     Buckets            Number of buckets.  Does not include
                                 the 'overflow' bucket
     2     Parameter-1        } Parameter use depends
   3-4     Parameter-2        } on distribution-valued
   5-6     Parameter-3        } attribute
```

For example, experiments with NeTraMet have used the following rules:

```
  FromPacketSize     & 1.0.25!1500 = 60.0!0:   PushPkttoAct, Next;

  ToInterArrivalTime &  2.3.1!1800 = 60.0.0!0: PushPkttoAct, Next;

  FromBitRate        & 2.3.1!10000 = 60.5.0!0: PushPkttoAct, Next;
```

In these mask and value fields a dot indicates that the preceding
number is a one-byte integer, the exclamation marks indicate that the
preceding number is a two-byte integer, and the last number is two
bytes wide since this was the width of the preceding field. (Note
that this convention follows that for IP addresses - 130.216 means
130.216.0.0).

The first rule specifies that a distribution of packet sizes is to be
built. It uses an array of 60 buckets, storing values from 1 to 1500
bytes (i.e. linear steps of 25 bytes each bucket). Any packets with
size greater than 1500 will be counted in the 'overflow' bucket,
hence there are 61 counters for the distribution.

The second rule specifies an interarrival-time distribution, using a
logarithmic scale for an array of 60 counters (and an overflow
bucket) for rates from 1 ms to 1.8 s. Arrival times are measured in
microseconds, hence the scale factor of 3 indicates that the limits
are given in milliseconds.

The third rule specifies a bit-rate distribution, with the rate being
calculated every 5 seconds (parameter 1). A logarithmic array of 60
counters (and an overflow bucket) are used for rates from 1 kbps to
10 Mbps. The scale factor of 3 indicates that the limits are given
in thousands of bits per second (rates are measured in bps).

These distribution parameters will need to be stored in the meter so
that they are available for building the distribution. They will
also need to be read from the meter and saved together with the other
flow data.

3.3 Reading Distributions

Since RTFM flows are bi-directional, each distribution-valued
quantity (e.g. packet size, bit rate, etc.) will actually need two
sets of counters, one for packets travelling in each direction. It
is tempting to regard these as components of a single 'distribution',
but in many cases only one of the two directions will be of interest;
it seems better to keep them in separate distributions. This is
similar to the old-style counter-valued attributes such as toOctets
and fromOctets.

A distribution should be read by a meter reader as a single,
structured object. The components of a distribution object are:

   - 'mask' and 'value' fields from the rule which created the
      distribution

   - sequence of counters ('buckets' + overflow)

These can be easily collected into a BER-encoded octet string, and
would be read and referred to as a 'distribution'.

4 Extensions to the Rules Table, Attribute Numbers

The Rules Table of "old-style" attributes will be extended for the
new flow types. A list of actions, and keywords, such as
"ToBitRate", "ToPacketSize", etc. will be developed and used to
inform an RTFM meter to collect a set of extended values for a
particular flow (or set of flows).

Note: An implementation suggestion.

   Value 65 is used for 'Distributions', which has one bit set for
   each distribution-valued attribute present for the flow, using bit
   0 for attribute 66, bit 1 for attribute 67, etc.

Here are ten possible distribution-valued attributes numbered
according to RTFM WG consensus at the 1997 meeting in Munich:

    ToPacketSize(66)   size of PDUs in bytes (i.e. number
    FromPacketSize(67)   of bytes actually transmitted)

```
      ToInterarrivalTime(68)   microseconds between successive packets
      FromInterarrivalTime(69)   travelling in the same direction

      ToTurnaroundTime(70)     microseconds between successive packets
      FromTurnaroundTime(71)     travelling in opposite directions

      ToBitRate(72)            short-term flow rate in bits per second
      FromBitRate(73)            Parameter 1 = rate interval in seconds

      ToPDURate(74)            short-term flow rate in PDUs per second
      FromPDURate(75)            Parameter 1 = rate interval in seconds

      (76 .. 97)               other distributions
```

   It seems reasonable to allocate a further group of numbers for the
   IIS attributes described above:

```
      QoSService(98)
      QoSStyle(99)
      QoSRate(100)
      QoSSlackTerm(101)
      QoSTokenBucketRate(102)
      QoSTokenBucketSize(103)
      QoSPeakDataRate(104)
      QoSMinPolicedUnit(105)
      QoSMaxPolicedUnit(106)
```

   The following attributes have also been implemented in NetFlowMet, a
   version of the RTFM traffic meter:

```
      MeterID(112)      Integer identifying the router producing
                           NetFlow data (needed when NetFlowMet takes
                           data from several routers)
      SourceASN(113)    Autonomous System Number for flow's source
      SourcePrefix(114) CIDR width used by router for determining
                           flow's source network
      DestASN(115)      Autonomous System Number for flow's destination
      DestPrefix(116)   CIDR width used by router for determining
                           flow's destination network
```

   Some of the above, e.g. SourceASN and DestASN, might sensibly be
   allocated attribute numbers below 64, making them part of the 'base'
   RTFM meter attributes.

   To support use of the RTFM meter as an 'Edge Device' for implementing
   Differentiated Services, and/or for metering traffic carried via such
   services, one more attribute will be useful:

      DSCodePoint(118) DS Code Point (6 bits) for packets in this flow

   Since the DS Code Point is a single field within a packet's IP
   header, it is not possible to have both Source- and Dest-CodePoint
   attributes. Possible uses of DSCodePoint include aggregating flows
   using the same Code Points, and separating flows having the same
   end-point addresses but using different Code Points.

5 Security Considerations

   The attributes considered in this document represent properties of
   traffic flows; they do not present any security issues in themselves.
   The attributes may, however, be used in measuring the behaviour of
   traffic flows, and the collected traffic flow data could be of
   considerable value. Suitable precautions should be taken to keep
   such data safe.

6 References

   [C-B-P]   Claffy, K., Braun, H-W, Polyzos, G., "A Parameterizable
                Methodology for Internet Traffic Flow Profiling," IEEE
                Journal on Selected Areas in Communications, Vol. 13, No.
                8, October 1995.

   [GUAR-QOS]   Shenker, S., Partridge, C. and R. Guerin, "Specification
                of Guaranteed Quality of Service", RFC 2212, September
                1997.

   [IIS-ACCT]   Maiocchi, S: "NeTraMet & NeMaC for IIS Accounting:
                Users' Guide", CEFRIEL, Milan, 5 May 1998. (See also
                http://www.cefriel.it/ntw)

   [IIS-RSVP]   Wroclawski, J., "The Use of RSVP with IETF Integrated
                Services", RFC 2210, September 1997.

   [IPPM-FRM]   Paxson, V., Almes, G., Mahdavi, J. and Mathis, M.,
                "Framework for IP Performance Metrics", RFC 2330, May
                1998.

   [RMON-MIB]   Waldbusser, S., "Remote Network Monitoring Management
                Information Base", RFC 1757, February 1995.

   [RMON2-MIB] Waldbusser, S., "Remote Network Monitoring Management
                Information Base Version 2 using SMIv2", RFC 2021,
                January 1997.

   [RTFM-ARC]   Brownlee, N., Mills, C. and G. Ruth, "Traffic Flow
                Measurement: Architecture", RFC 2722, October 1999.

   [RTFM-MIB]   Brownlee, N., "Traffic Flow Measurement: Meter MIB", RFC
                2720, October 1999.

7 Authors' Addresses

Sig Handelman
IBM Research Division
T.J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Phone: +1 914 784 7626
EMail: swhandel@us.ibm.com

Stephen Stibler
IBM Research Division
T.J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Phone: +1 914 784 7191
EMail: stibler@us.ibm.com

Nevil Brownlee
Information Technology Systems & Services
The University of Auckland
Private Bag 92-019
Auckland, New Zealand

Phone: +64 9 373 7599 x8941
EMail: n.brownlee@auckland.ac.nz

Greg Ruth
GTE Internteworking
3 Van de Graaff Drive
P.O. Box 3073
Burlington, MA 01803, U.S.A.

Phone: +1 781 262 4831
EMail: gruth@bbn.com

8. Full Copyright Statement

Acknowledgement