Network Working Group   Y. Kikuchi
Internet-Draft   Kochi University of Technology
Intended status: Standards Track   S. Matsushima
Expires: November 22, 2008   Softbank Telecom Corp.
                                                          K. Nagami
                                                   Intec Netcore Inc.
                                                             S. Uda
                                          Japan Advanced Institute of
                                               Science and Technology
                                                        N. Ogashiwa
                                       Maebashi Kyoai Gakuen College
                                                        May 21, 2008

              One-way Passive Measurement of End-to-End Quality
                   draft-kikuchi-passive-measure-02.txt

Status of this Memo

Abstract

   This draft describes a passive measurement method for one-way end-to-
   end quality. This method reports both whether a flow of packets is
   in-sequence or not and error types on detecting out-of-sequence. The
   method consumes small resource therefore it can be adapted to many
   protocols that have sequence number field.

Table of Contents

1. Introduction

   This draft describes a passive measurement method for one-way end-to-
   end connections quality. This method reports both

   o whether a flow of packets is in-sequence or out-of-sequence, and

   o error types on detecting out-of-sequence.

   The purpose of the measurement by this method is to maintain
   transports in operation [I-D.kikuchi-tunnel-measure-req] so that the
   algorithm was designed to be lightweight. The algorithm detects out-
   of-sequence packets strictly though; the error metrics is even not
   accurate because the metrics should only provide some hints to
   operators.

   The algorithm requires a sequence number field in the packet headers,
   such as extended GRE[RFC2784] [RFC2890], PWE3[RFC3985] and
   RTP[RFC3550].

1.1. Requirements notation

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in [RFC2119].

2. **Metrics**

   We firstly define ``in-sequence'' of a flow.

   o in-sequence: the order of arrival packets at the egress of a flow
      is the same order as departure at the ingress.

   o out-of-sequence: a packet exists, which violates in-sequence state
      of a flow.

   We secondly define the metrics of types of out-of-sequence packet.

   o dup-train: packet that is identical to the immediately preceding
      arrival packet

   o skipping: packet that should arrive next or more lately

   o astern: packet that arrives after the successive packets' arrival

   The metrics ``dup-train'', ``skipping'' and ``astern'' are similar to
   duplication, lost and reordering respectively though, these metrics
   above have different names because the definitions are slightly
   different from any metrics defined before.

   o ``dup-train'' does not mean naive duplication because the pair of
      dup-train packets should arrive uninterruptedly by another packet.

   o ``skipping'' happens on ``packet loss'' and even by
      ``reordering''.

   o ``astern'' might count inaccrately for as reordering packets on
      complicated situation.

   This kind of difference comes from the request of let the algorithm
   lightweight described in Section 3. An accurate algorithm must have
   much space for keeping the order of arrival packets until the exact
   irregular reason will be become clear, moreover it takes much
   computing power according to the space consumption. Those rough
   metrics above allow to shrink the space that keeps just one sequence
   number of the previously arrival packet.

   Note that we do not define some kinds of ``rates'' because they are
   easy to derived from total numbers of irregular packets. For
   example, if you use SNMP [RFC3411] the rates are calculated from
   periodical polling by an SNMP manager.

3. Measurement Method

   In this section, we illustrate a method to measure qualities defined
   in the previous section. The protocol should have the sequence
   number field in its headers.

3.1. Counters and Functions

   Egress router of a flow must have a register, named ``recvseq'', that
   maintains the number that the successive arriving packet should have.
   The ``recvseq'' register should be initialized by the specific number
   depends on the target protocol.

   The router must also have the following counters:

   o duptrcnt: the number of dup-train packets,

   o skipcnt: the number of skipping packets, and

   o astrncnt: the number of astern packets

   Let the counters above be unsigned integer and initialized by 0 on
   the birth of the flow. The length of the counters should be the same
   as the sequence number field defined in the protocol.

3.2. Measurement Algorithm

   This algorithm determines whether a receiving packet is normal or not
   while comparing a counter "recvseq" with the sequence number of the
   packet named "seqno". The basic idea consists of the following
   conditions.

   1. if recvseq and seqno are same then "in-sequence",

   2. else if seqno is just a predecessor of recvseq then "dup-train";

   3. otherwise if seqno proceed then "skipping" else "astern".

   The following C-like codes specifies the algorithm in detail. The
   function measure will be invoked by every one of the receptions of
   packets.

```
boolean measure(packet_t packet)
{
        unsigned int seqno;
        seqno = packet->header->seqno;  // getting seq # from header

        if (seqno == recvseq) {
                recvseq++;
                return true;             // in-sequence
        } elsif (seqno+1 == recvseq) {  // same # as predecessor's
                duptrcnt++;              // determines a dup-train
                return false;            // out-of-seq by dup-train
        }

        signed int diff;
        diff = (signed int)(seqno - recvseq);
        if (diff > 0) {                  // from past or future?
                skipcnt += diff;         // determines a skipping
                recvseq = seqno;
        } else {                         // means it is a past packet
                astrncnt++;              // determines a astern
        }
        return false;                    // out-of-seq w/o dup-train
}
```

Figure 1

The function ``measure'' returns true only if the packet is in-
sequence otherwise false on out-of-sequence. The value can be used
to discard the packet when the protocol does not allow passing
irregular packets to its higher layer.

4. Requirements to Sequence Numbering

   In this section, we describe the requirements for protocols to adopt
   this method.

4.1. Indication of Sequence Number

   The protocol MUST indicate whether sequence numbering is enabled or
   not.

   There are two ways to indicate whether the sequence numbers are
   enabled or not. One is to prepare an indication field in the header
   independent from the sequence number field.

   The other is to indicate a special sequence number, typically 0,
   meaning disabled. In this case, the measurement process needs
   additional steps on wrapping sequence number overflow because the
   sequence number will skip 0 that does not seem continuous even if the
   flow packets are still in-sequence.

4.2. Field Length

   The length of sequence number field SHOULD be long enough according
   to the transmission speed. Otherwise, the period of a lap of the
   sequence number becomes too short and the reliability of the
   measurement decreases.

   For example, the algorithm may determine packets loss as reordering,
   when there is a set of burst packets loss in case of the path change.
   It is necessary to determine whether a burst packet loss occurred or
   if it was simply the arrival of a very past packet when the
   difference of the sequence numbers between two continuous packets is
   very large. The typical technique is to use half of the
   representable maximum value. This is simple and adequate if the
   field is long enough.

   However, the existence of the sequence number field generates more
   amounts of transmission packets. Thus, if an insufficiently long
   field creates overhead for protocols that are sensitive to resource
   consumption. The sequence number field length should be considered
   as a tradeoff between bandwidth efficiency and quality assurance.

5. An Evaluation Trial

Here we try to evaluate the metrics according to the framework
described in [I-D.morton-perf-metrics-framework].

5.1. Usefullness

Firstly, we check the usefullness of the metrics listed in the
section 3.1 of [I-D.morton-perf-metrics-framework].

1. The degree to which its absence would cause significant loss of
    information on the behavior or state of the application or system
    being measured?

    * YES for TSPs who have SLA contracts with their customers.

    * NO for best effort oriented flows.

2. The correlation between the metric and the quality of service /
    experience delivered to the user (person or other application)?

    * YES. It shows primitive quality to determine a flow traffic
       in operation.

    * NO. There is no correlation to any application because the
       metrics are independent from specific applications.

3. The degree to which the metric is able to support the
    identification and location of problems affecting service
    quality?

    * YES. It helps for operators to find out that problems come
       from whether application or transport.

    * NO. It does not indicate that problems come from whether TSP
       or intermediate ISPs.

5.2. Qualifying Metrics

Secondly, we qualify the metrics by the list in the section 3.5 of
[I-D.morton-perf-metrics-framework].

o Unambiguously defined? Yes. The description is not by a concrete
   specification in natural language but in C-code.

o Units of Measure Specified? Yes.

o Measurement Errors Identified? Refer to Section 4.2 for less
   sequence field length.

o Repeatable? Yes.

o Implementable? Yes, very much.

o Assumptions concerning underlying process? Not discussed.

o Use cases? Yes. Some examples are shown in Section 6. Moreover,
   there is an implementation using the metrics in a multi-homing
   solution with [RFC4908], which has been provided by Intec Netcore
   Inc.

o Correlation with application performance / user experience? The
   discussion is done in Section 5.1.

5.3. Reporting Model

The document does not define any reporting model for the metric.

6. **Algorithm Behavior**

   The following diagrams show the behavior of the algorithm on
   receiving out-of-sequence packets. Figure 3 shows the legend of flow
   diagram here. The left and right sides represent the sender and
   receiver of a flow respectively. Time flows upper to lower in the
   diagrams. This illustrates a normal transmission with the sequence
   number n.

```
   time            sender                 receiver

     |             |                      |
     |         n   |                      |   n    0 0 0
     |             |----[seq #: n]----->|
     |       n+1   |                      | n+1  0 0 0
     |             |                      |
     V             V                      V
```
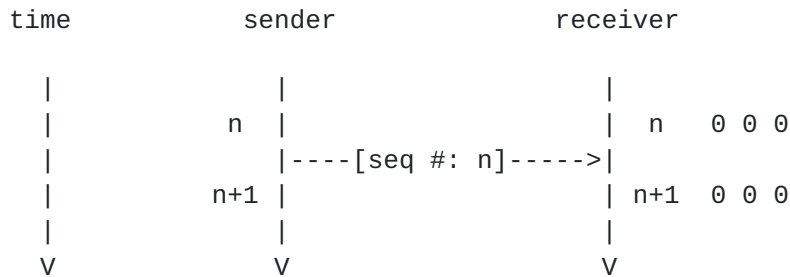
                         Figure 2

   Figure 3, Figure 4 and Figure 5 show simple cases, such as loss,
   duplication and reordering of packets respectively.

```
                 [astrncnt]................
                 [duptrcnt].............. :
                  [skipcnt]............ : :
                                       : : :
                 [recvseq].........  : : :
           ........[sendseq]        :   : : :
          :                         :   : : :
          :                         :   : : :
          :                         :   : : :
            |                         |
        0 |                         | 0   0 0 0
            |------------------>|
        1 |                         | 1   0 0 0
            |------------------>|
        2 |                         | 2   0 0 0
            |-----> !LOST!        |
        3 |                         |
            |------------------>|
        4 |                         | 4   1 0 0
            |-----> !LOST!        |
        5 |                         |
            |-----> !LOST!        |
        6 |                         |
            |------------------>|
        7 |                         | 7   3 0 0
            |                         |
             V                       V
```

                      Figure 3

```
                [astrncnt]................
                [duptrcnt].............. :
                 [skipcnt]............ : :
                                     : : :
                [recvseq]........  : : :
         ........[sendseq]         :  : : :
         :                         :  : : :
           |                   |
         0 |                   | 0  0 0 0
           |------------------>|
         1 |                   | 1  0 0 0
           |-------!DUP!------>|
           |          \        | 2  0 0 0
           |           \------>|
         2 |                   | 2  0 1 0
           |------------------>|
         3 |                   | 3  0 1 0
           |-------!DUP!------>|
         4 |          \        | 4  0 1 0
           |        !DUP!----->|
           |            \      | 4  1 2 0
           |             \---->|
           |                   | 4  1 3 0
           |------------------>|
         5 |                   | 5  1 3 0
           |                   |
           V                   V
```

                      Figure 4

```
                    [astrncnt]...............
                    [duptrcnt]............. :
                     [skipcnt]........... : :
                                         : : :
                    [recvseq].........  : : :
            ........[sendseq]          :  : : :
            :                          :  : : :
             |                    |
         0 |                    | 0   0 0 0
             |------------------>|
         1 |                    | 1   0 0 0
             |-------\          |
         2 |          \         |
             |---------\-------->|
         3 |           \        | 3   1 0 0
             |            \------->|
             |                    | 3   1 0 1
             |------------------>|
         4 |                    | 4   1 0 1
             |-------\          |
         5 |          \         |
             |------\  \         |
         6 |        \  \        |
             |--------\--\------->|
         7 |          \  \      | 7   3 0 1
             |            \   \----->|
             |              \      | 7   3 0 2
             |               \------>|
             |                    | 7   3 0 3
             |                    |
             V                    V
```
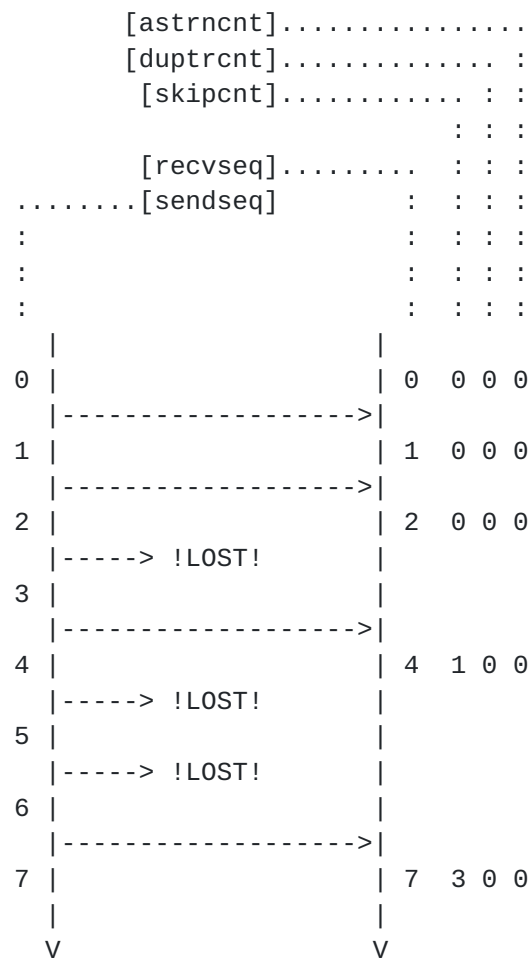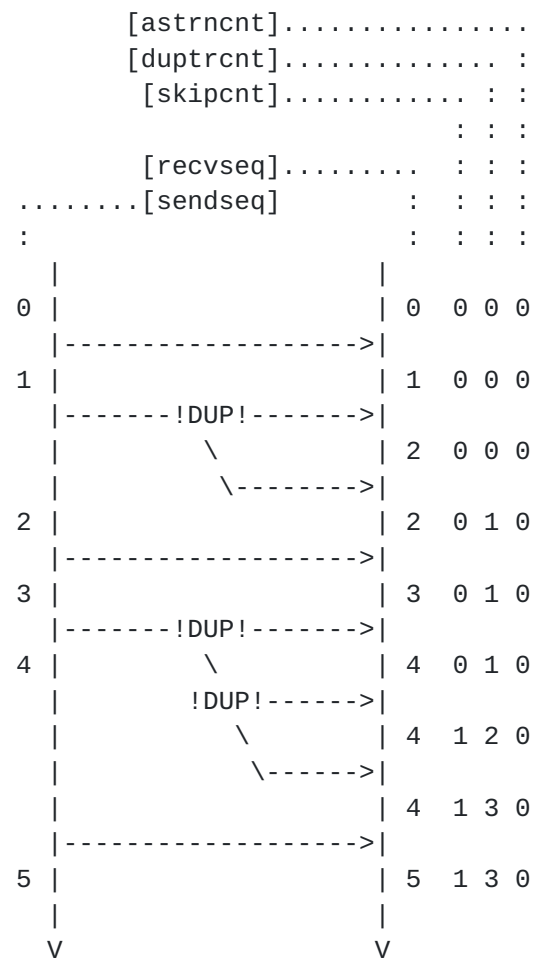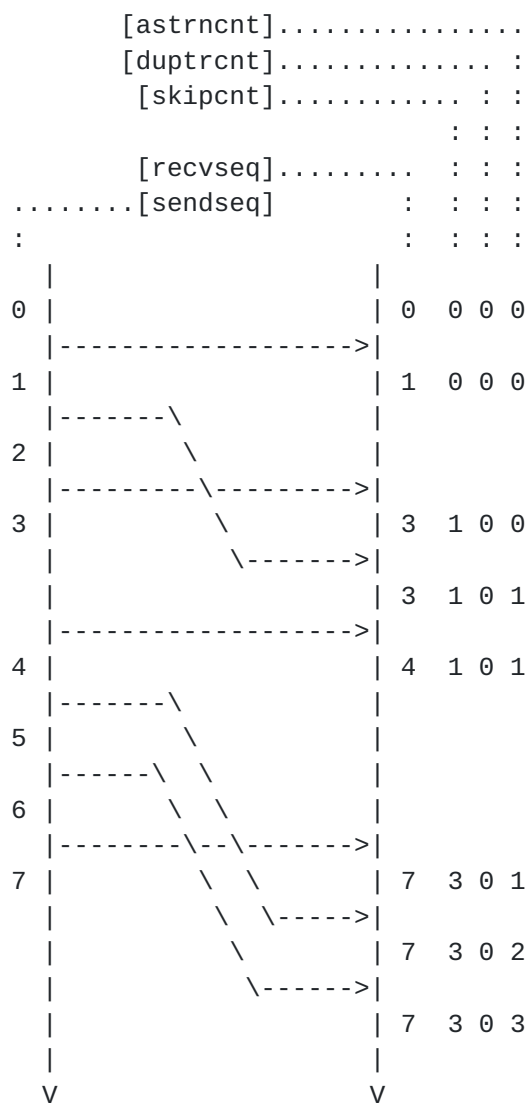
                        Figure 5

   Figure 6 and Figure 7 show cases in a little bit complex situations.
   Figure 6 shows that the algorithm cannot distinguish a combination of
   duplication and loss from a reordering. Compare the flow to former
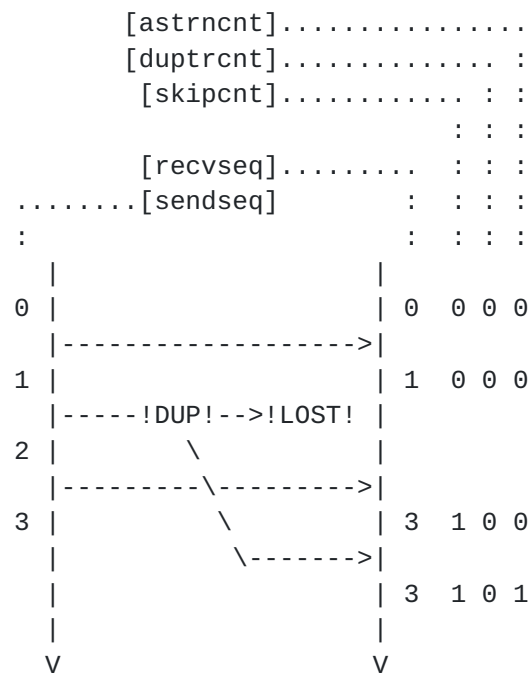   of Figure 5.

```
                 [astrncnt]...............
                 [duptrcnt]............. :
                  [skipcnt]........... : :
                                      : : :
                 [recvseq].........  : : :
            ........[sendseq]       :  : : :
            :                       :  : : :
               |               |
         0 |               | 0  0 0 0
           |----------------->|
         1 |               | 1  0 0 0
           |-----!DUP!-->!LOST! |
         2 |        \          |
           |---------\-------->|
         3 |          \        | 3  1 0 0
           |             \------->|
           |               | 3  1 0 1
           |               |
           V               V
```

                            Figure 6

   Figure 7 shows that the algorithm interprets duplication as
   reordering when a duplicated packet does not arrive in succession.
   It is not possible to hold all of the information contained in the
   arrival packets needed to measure accurately.

```
                 [astrncnt]...............
                 [duptrcnt]............. :
                  [skipcnt]........... : :
                                      : : :
                 [recvseq].........  : : :
            ........[sendseq]       :  : : :
            :                       :  : : :
               |               |
         0 |               | 0  0 0 0
           |----------------->|
         1 |               | 1  0 0 0
           |------!DUP!-------->|
         2 |        \          | 2  0 0 0
           |----------\------->|
         3 |          \        | 3  0 0 0
           |             \------->|
           |               | 3  0 0 1
           V               V
```
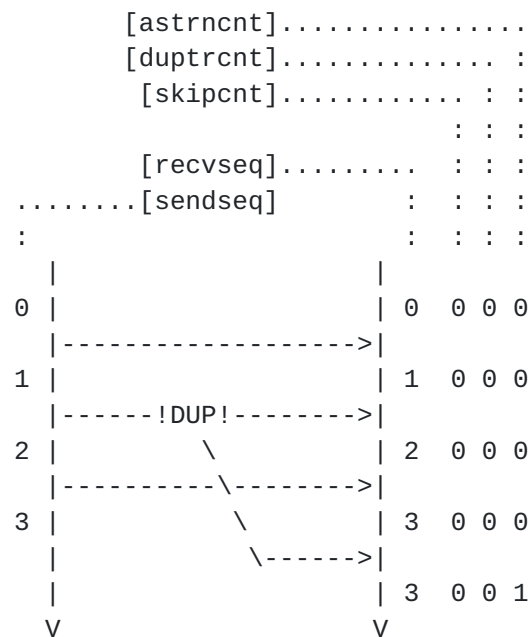
                            Figure 7

7. Security Considerations

   The passive measurements do not use any additional packets and flows,
   so that most security considerations boil down to the issues of the
   original protocols. For example, fraud sequence numbers cause the
   measurement process to become disorganized. This discussion boils
   down to the issues of the header protection.

Appendix A. Acknowledgements

   The authors would like to thank for helpful discussions in TEReCo 2.0
   research project sponsored in part by the ministry of internal
   affairs and communications Japan (SCOPE 072309007).

8. References

8.1. Normative References

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119, March 1997.

8.2. Informative References

   [I-D.kikuchi-tunnel-measure-req]
              Kikuchi, Y., Matsushima, S., Nagami, K., and S. Uda,
              "Quality Measurement Requirements for Tunneling
              Protocols", draft-kikuchi-tunnel-measure-req-02 (work in
              progress), November 2007.

   [I-D.morton-perf-metrics-framework]
              Morton, A., "Framework for Performance Metric
              Development", draft-morton-perf-metrics-framework-00 (work
              in progress), October 2007.

   [RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003,
              October 1996.

   [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P.
              Traina, "Generic Routing Encapsulation (GRE)", RFC 2784,
              March 2000.

   [RFC2890] Dommety, G., "Key and Sequence Number Extensions to GRE",
              RFC 2890, September 2000.

   [RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An
              Architecture for Describing Simple Network Management
              Protocol (SNMP) Management Frameworks", STD 62, RFC 3411,
              December 2002.

   [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.
              Jacobson, "RTP: A Transport Protocol for Real-Time
              Applications", STD 64, RFC 3550, July 2003.

   [RFC3985] Bryant, S. and P. Pate, "Pseudo Wire Emulation Edge-to-
              Edge (PWE3) Architecture", RFC 3985, March 2005.

   [RFC4908] Nagami, K., Uda, S., Ogashiwa, N., Esaki, H., Wakikawa,
              R., and H. Ohnishi, "Multi-homing for small scale fixed
              network Using Mobile IP and NEMO", RFC 4908, June 2007.

Authors' Addresses

    Yutaka Kikuchi
    Kochi University of Technology
    306B Research Collaboration Center
    185 Miyanokuchi, Tosayamada-cho
    Kami-shi, Kochi 782-0003
    JP

    Email: yu@kikuken.org

    Satoru Matsushima
    Softbank Telecom Corp.
    1-9-1 Higashi-Shinbashi, Minato-ku
    Tokyo
    JP

    Email: satoru@ft.solteria.net

    Ken-ichi Nagami
    Intec Netcore Inc.
    1-3-3, Shin-suna
    Koto-ku, Tokyo
    JP

    Email: nagami@inetcore.com

    Satoshi Uda
    Japan Advanced Institute of Science and Technology
    1-1 Asahi-dai
    Nomi-shi, Ishikawa-ken 923-1292
    JP

    Email: zin@jaist.ac.jp

    Nobuo Ogashiwa
    Maebashi Kyoai Gakuen College
    Koyahara 1154-4
    Maebashi, Gunma 379-2192
    JP

    Email: ogashiwa@c.kyoai.ac.jp