

Passive One-Way-Delay Measurements and Data Export

Tanja Zseby, Lutz Mark, Carsten Schmoll, Guido Pohl

Fraunhofer FOKUS

Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany

{zseby, mark, schmoll, pohl}@fokus.fraunhofer.de

Abstract

This document describes a non-intrusive method for measuring one-way delay of IP packets. Furthermore it describes how to use NetFlow Version 9 to export the data of such measurements and evaluation processes.

1. Introduction

Motivations for One-way delay measurements are listed in RFC2679.

There exist a variety of motivations for performing passive measurements in IP networks. Many applications require the measurement of the Quality of Service (QoS) for the transport of specific IP flows or traffic aggregates. Network providers and customers are interested whether negotiated QoS values in SLAs are met (SLA validation). Measurements also provide the basis for usage-based accounting. Furthermore, measurement results are an important input for traffic engineering decisions.

2. Terminology

IP Packet Capturing Process

An IP packet selection process takes IP packets or parts of IP packets (e.g. header) as input and extracts a subset of these packets by applying a selection function.

Filtering

Filtering selects a subset of packets by applying deterministic functions on parts of the packet content like header fields or parts of the payload.

Metering process

see definition in [QuZC02]

3. General Architecture

Passive one-way delay measurements require two measurement points. Both measurement points generate a timestamp and a unique packet ID for each packet and send this information to a control instance that calculates the one-way delay. The packet ID is needed to associate the timestamps from the different measurement points to the correct packet.

For each packet that is measured a timestamp and a packet ID have to be generated, stored and transmitted to a collection point where the QoS computation takes place, based on the results from the different measurement points. Therefore the amount of measurement result data that arises per second depends on the number of measured packets n per second, the number of bits l_t used for the representation of the timestamp and the number of bits l_{id} for the packet ID.

The main building blocks for implementing a one-way-delay measurement are shown in Figure 1.

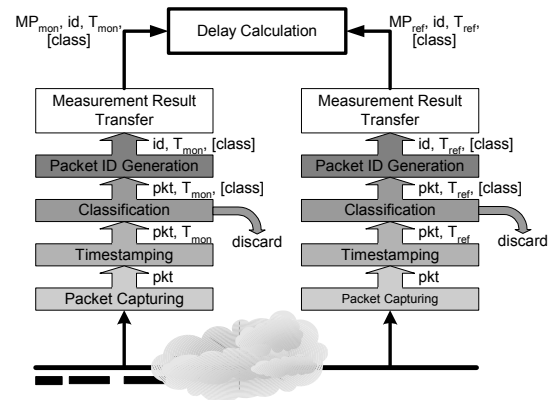


Figure 1: OWD Measurement Components

The processes involved are packet capturing, timestamping, classification, generation of a packet ID and transfer of measurement data. The requirements for the building blocks are examined in detail in the following subsections. Based on these building blocks, we investigate alternative methods for implementing them, in order to identify the most efficient way to perform the needed tasks.

Further issues that must be dealt with are privacy issues when capturing traffic from customers, difficulties in packet event correlation when packets are lost or duplicated, and the overall amount of measurement data captured and transmitted for evaluation.

In order to measure the end-to-end delay meter must be placed accordingly, which means as close to the end points.

4. Packet Capturing

A certain amount of bytes needs to be captured per packet as basis for the generation of a packet ID. The packet ID collision probability depends on the generation function and the number of bytes that are used as input. In [DuGr00] it is stated that the first 40 Bytes starting at the IP header are sufficient.

5. Timestamping

A number of issues have to be considered for the basic function of assigning timestamps to packets for subsequent delay calculation. Internal buffering in the hardware and on the way through the kernel causes additional packet delay. Even if all involved measurement devices are equipped with the same hardware and operating system, packets can experience different delays (e.g. due to CPU load and the level of buffer filling). In order to reduce effects from additional variable delays, the timestamp should be assigned to the packet as early as possible.

A further problem that has to be solved when using two measurement points is clock synchronization between both points. Current solutions are based on the Network Time Protocol (NTP) [RFC1305], the Global Positioning System (GPS), and radio signals (e.g. DCF77). Each solution has its own drawbacks and advantages.

A timestamp can be represented as absolute time. With this the number of bits needed for the representation of the timestamp depends only on the desired accuracy for the measured metric. A possibility to reduce the number of bits l_t used for the timestamp is to use relative timestamps. One approach is to make an assumption on the maximum time t_{max} a packet needs to traverse the network from the ingress to the egress measurement point. With this upper limit the timestamp needs to be non-ambiguous only within this limit. In this case the value l_t depends not only on the desired accuracy of the time representation but also on the predetermined limit for the maximum time the packet needs to traverse the network. Another possibility is to use an absolute timestamp only for the first packet in a given time interval $[0, t_{int}]$ and use timestamps relative to this for successive packets that arrive in the same interval.

Further issues are that the time that is needed for the timestamping process can differ for subsequent packets, which can also lead to inaccuracy [CIDG00].

6. Filtering / Classification

A classification of packets is required if only selected packets are used for the measurement. A pre-selection is useful to reduce the amount of resulting measurement data and the required processing time for the subsequent processes (like packet ID generation). It could be also considered to place the classification before the timestamping in order to relieve the timestamping process. But this would contradict the requirement to do the timestamping as early as possible in the chain. Therefore the pros and cons have to be examined carefully in advance before changing the chain sequence shown in Figure 1.

The classification can filter out packets with specific characteristics. These can be for example all packets that belong to a specific flow or traffic aggregate (characterized by a common DiffServ Codepoint) to determine the quality for specific applications or traffic classes.

In certain cases it is important to maintain the information to which flow or class the measured packet belongs to. For instance if the quality for different DiffServ traffic aggregates is measured simultaneously it is desired to keep the information about the class together with packet ID and timestamp. In some cases the packet ID already contains additional information because it has been calculated with a bijective function on the fields of information (e.g. a lossless compression function that compresses the IP header can be decompressed to determine the information on source and destination). In all other cases, the wanted information has to be transferred to the analysis application in addition to the packet ID. Since the packet ID provides a unique mapping to the packet, this additional information does only need to be transferred at one measurement point. The other measurement point can determine the information via mapping its packet ID to the information stored under the same packet ID at the other measurement point.

7. Packet ID Generation

Unlike semi-active measurements, passive measurements are based on methods where packets are neither marked nor modified in another way. Therefore the recognition has to be based on fields that already exist in the packet. In order to get the same packet ID for one packet at both measurement points the packet ID generation should be based only on fields that are invariant or predictable during the transport. Fields that are highly variable between the packets (e.g. the datagram ID for IPv4) are more suitable than fields that are nearly constant or vary only between a few values (e.g. version field). The generation of a packet ID should be based on

fields that already exist in the packet (so no modification of the packet is required), are invariant or predictable during the transport (at least on the path from the ingress to the egress second measurement point) and are highly variable between the different packets.

The request for low collisions (uniqueness of the ID) contradicts the request for a small packet ID; because the more bits are used for representing the packet ID the lower is the probability of collisions. The collision probability within a traffic trace depends on

- the distribution of the bit sequences taken as input to the packet ID generation (that means it is highly dependent on the considered traffic mix),
- the packet ID generation function,
- the size of the packet ID lid,
- the used Operating System (OS) (if the datagram ID is considered)

The goal is to achieve an acceptable low probability of collisions with a packet ID that does not exceed the available capacity for the measurement result data transfer. As for the timestamp, the packet ID only needs to be unique in the given time interval $[0, t_{max}]$. This limits the number of possible combinations to the number of packets n_{max} that can be observed within this interval. For example for a 155 Mbits/s link with an average packet size of 512 Bytes and a maximum time to traverse the network of 10s, n_{max} would be 378,420 packets. 19 bits can represent this amount of combinations.

Ipv4 and Ipv6 packets have different packet headers. That implies that the fields of the IP packets, which can be used for packet ID generation differ between Ipv4 and Ipv6 packets.

Ipv4: ip-len, ip-id, ip-prot, srcaddr, dstaddr, X bytes payload

Ipv6: payload-len, ip-prot, srcaddr, dstaddr, Y bytes payload

There are different possibilities to generate an id from the considered fields:

- unprocessed plain
- one-way hash function (e.g. MD5)
- checksum CRC
- compression function

Functions that map a large bit sequence (the selected fields of the packet) to a smaller bit sequence (the packet id) can always increase the collision probability. Using the selected fields of the packet without further processing means to use the calculation basis itself instead of a derived id. This would lead to the minimum collision probability that ever can be achieved with the given traffic mix. Furthermore it would reduce the

required processing power because no function has to be performed on the selected fields. Nevertheless this method would result in a packet id size m that is equal to the sum of bits in the selected fields. Especially for small packets this would increase the rate required for the measurement report messages up to the rate for the data flow itself.

Despite of several ID generation schemes, we exclusively propose to use CRC32.

8. OWD Calculation Process

To estimate the one-way delay of an IP packet between two monitor points Ref and Mon the difference of the arrival times of the packet at the two monitor points is calculated:

$$T_{owd} = T_{ref} - T_{mon}$$

The correlation between the two timestamps to the same IP packet is done via the packet ID and a T_{delta} . If a packet of a special ID is captured at the reference point but not detected at the monitor point within T_{delta} the packet is considered as lost.

9. Measurement Result Transfer

In order to calculate QoS parameters like delay, two timestamps have to be compared. If more than one measurement point is involved the measurement results (timestamps and packet ID) from the different measurement points have to be collected at a common location in order to calculate the delay value. This collection point can be located on a separate host. It also can be co-located with one of the meters in order to reduce the amount of data that has to be transferred. The following possibilities to transfer the measurement results have to be distinguished:

In-packet: The measurement results (timestamps and packet ID) from the first measurement point for packet recognition and the timestamp are carried within the packet.

In-band: The measurement results are sent directly on the same path as the data.

Out-of-band: The measurement results are sent on a separate path.

Solution a) requires the modification of packets. This would lead to a semi-active measurement method with all its disadvantages (e.g. the need for packet modification at link rate). The advantage is that the analysis can take place directly at the second measurement point. This

method is efficient especially if packets already contain information that can be used for the packet event correlation (e.g. an IPsec authentication header).

Solution b) leads to additional load on the network under test. That means measurement result data packets can influence the original data flow and can lead to the same disadvantages that active measurements observe. The in-band sending of packets with measurement results therefore somehow contradicts to the passive approach where no influence on the original traffic is desired. It could be seen as a special form of semi-active measurements, where measurement data is sent in separate packets instead of including the information directly into the data packets (by modifying them). Nevertheless, there is a difference between sending test traffic for active measurements and the transmission of measurement results. The amount, type and timeframe for the sending of test traffic for active measurements are dictated by the measurement task. In contrast to this, the sending of measurement results can be controlled by other means. For instance it could be sent with a lower priority (e.g. lower-than-best-effort class) or only at times when the network is loaded lightly, or routed over paths that are currently not loaded (e.g. via MPLS). Which alternative is to be preferred depends on the policy for the evaluation of the metric (e.g. real-time or non-real-time).

Solution c) requires a separate path to the analysis application. This can be achieved for instance via a second interface card and a separate measurement network that connects all measurement points. This approach does not influence the data traffic but requires capacity in a different network.

In all three cases, additional capacity either on the existing network or on a separate network is required. For economic reasons even a separate reporting network would probably have a lower capacity than the “production network”. Therefore to save resources (storage capacity and bandwidth) the measurement result traffic should be kept as low as possible.

10. NetFlow Version 9 Measurement Result Export

NetFlow Version 9 is one candidate of the IPFIX working group examined to serve as a standardized exporting protocol for measurement data and flow information export.

As the name NetFlow Version 9 indicates, its purpose is to exchange information about network traffic flows. In this scope flow is defined by a handful of key attributes (source/destination address, source/destination port, Layer3 Protocol Type, TOS/DSCP byte, interface of the flow exporting network element).

In the initial sense, a flow aggregates a number of packets with common attributes. For passive One-Way-Delay measurements, however, it is necessary to step one level of abstraction back and export packet data. It is clear that one could consider a single packet as a special case of a flow. This procedure has consequences that conflict with the efficiency of the exporting procedure. When many packets belong to the same flow, i.e. they share common attributes; these attributes would have to be exported on a per-packet basis although the information is redundant.

In the following paragraphs we propose how to use NetFlow Version 9 for exporting packet information and OWD data.

10.1. Export of Packet ID and Timestamps

For Measurement Process Results we define two different template records, namely Packet Properties and Flow Properties.

As indicated in Figure 2, the Flow Properties template defines the attributes for a flow; exemplarily IP source and destination address. The important field is the Flow Index Field. As we will explain the Flow Index will allow packet records to reference flow attributes. The Flow Index is a unique identifier for flow definitions. Subsequent data records (marked as R1 ... R_k) define the actual flows. The FlowSet ID should indicate the special case of a Flow Property template. (FlowSet IDs 0 and 1 are already reserved for templates consisting of flow fields or option fields, respectively. However, the Collector must handle Flow Properties templates and their data records slightly different than usually.)

As for the information of a single packet that is defined in the Packet Properties template. This information is packet specific and normally not shared between *many* packets, as otherwise one would rather consider the information as flow related and therefore it has not to be exported for every packet.

The Packet Properties template consists of at least Timestamp and Packet ID. Additionally, this template contains a Flow Index field. In packet records, the value of this field will contain one of the unique indices of the flow records exported before.

Figure 2 shows three arbitrary packet records (R1 ... R_j). For clarification the linkage of packets and flows has been marked with vertices between the Flow Index values of packet records and flow records at the right margin of the picture. In the figure the flow record R1 of the Flow Properties is associated with packet record R_j. No packet records reference flow record R2. The flow record R_k is referenced twice, by packet record R1 and R2.

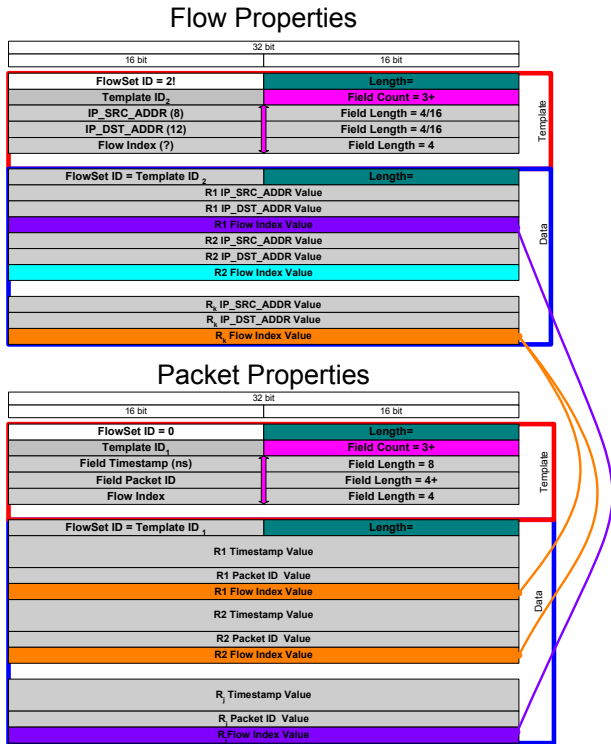


Figure 2: Flow and packet properties

10.2. Export of OWD data

At the collection point packet records of two measurement points are gathered and correlated by means of the Packet ID. The One Way Delay is then calculated as stated earlier. The resulting delay data records are exported in a similar manner as the packet data have been. Especially, the linkage between delay data and flow information is represented with the discussed Flow Index fields. The OWD Properties contain the Packet Pair ID (which is the Packet ID of the two contributing packet records), a Timestamp (which is the Timestamp of the packet passing the reference monitor point) in order to reconstruct a time scale, the calculated delay value, and finally a Flow Index.

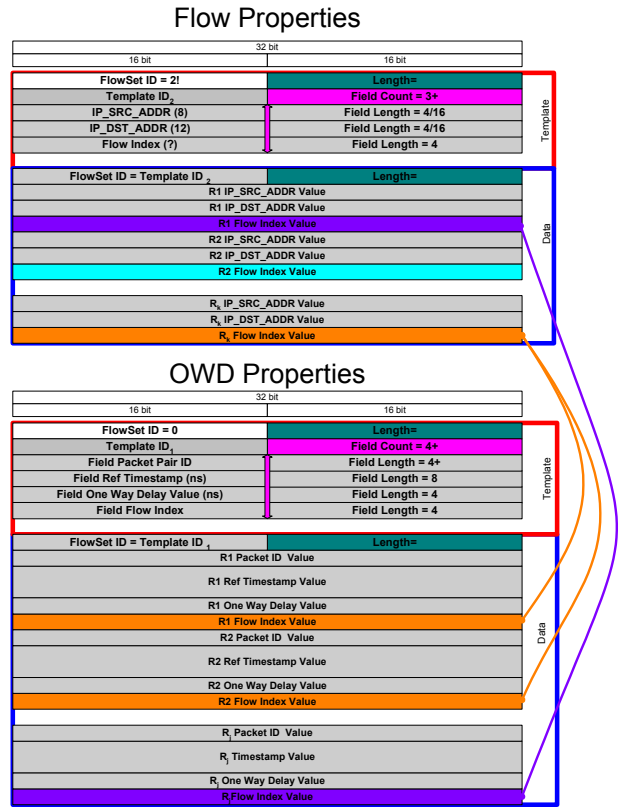


Figure 3: Flow and OWD properties

10.3. Export and Collection Considerations

In order to use NetFlow Version 9 for packet and OWD data export we have to prepare several things. We must assign a new FlowSet ID from the range 2 to 255 for the Flow Properties templates.

We have to introduce new Field Type definitions for

- Packet ID, Length N,
- Flow Index, Length 4
- Timestamp, Length 8
- One Way Delay, Length 4

The size of the Packet ID depends on the used generator function – in the case of a CRC32 the field size would be 4 bytes. The range for the Flow Indices should be reasonable large we propose a 32-bit value, i.e. 4 bytes.

We suggest using absolute timestamps with a resolution of nanoseconds and a width of 64-bit (signed). The origin will be Jan. 1, 1970 as usual. (The range covers ~292 years. So we provoke a Year-2262-Problem...)

The value for the One-Way-Delay should be a 32-bit (signed) value, which covers ~2 seconds.

A collecting instance must store, the flow records of Flow Properties together with the Flow Indices. The flow

records therefore should be handled like templates. A similar timeout and refresh mechanism as for templates should be used for flow records too (refer to [Clai02]).

The collector must be able to associate incoming packet records to flow records by matching contained Flow Indices.

For obvious reasons, the exporting process must send the appropriate defining flow records with the unique Flow Indices before it can send packet records that reference flow information by indices.

12. References

[DuGG02] Nick Duffield, Albert Greenberg, Matthias Grossglauser, Jennifer Rexford: *A Framework for Passive Packet Measurement*, Internet Draft draft-duffield-framework-papame-01, work in progress, February 2002

[DuGr00] Nick Duffield, Matthias Grossglauser: “*Trajectory Sampling for Direct Traffic Observation*”, Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, August 28 - September 1, 2000

[QuZC02] J. Quittek, T. Zseby, B. Claise, S. Zander, G. Carle, K.C. Norseth: *Requirements for IP Flow Information Export*, Internet Draft <draft-ietf-ipfix-reqs-05.txt>, work in progress, August 2002

[GrDM98] Ian D. GRAHAM, Stephen F. DONNELLY, Stele MARTIN, Jed MARTENS, John G. CLEARLY: *Nonintrusive and Accurate Measurement of Unidirectional Delay and Delay Variation on the Internet*, INET'98, Geneva, Switzerland, July 21-24, 1998

[ZsZC01] T. Zseby, S. Zander, G. Carle: *Evaluation of building blocks for passive one-way-delay measurements*, PAM2001, Amsterdam, April 23-24, 2001

[Clai02] B.Claise: *Cisco Systems NetFlow Services Export Version 9*, Internet Draft <draft-bclaise-netflow-9-00.txt>, work in progress, December 2002