Shall we worry about Packet Reordering?

Michal Przybylski (michalp@man.poznan.pl)
Bartosz Belter (bart@man.poznan.pl)
Artur Binczewski (artur@man.poznan.pl)

Poznan Supercomputing and Networking Center, ul. Noskowskiego 10, Poznań, POLAND
Tel: +48 61 8582027 Fax: +48 61 8525954

Keywords

Packet Reordering, QoS, GEANT, testbed, results, applications

Introduction

Packet reordering is yet another network Quality of Service (QoS) parameter that has to be taken into account in modern networking. This not clearly understood and a common phenomenon is affecting leading-edge broadband solutions such as those implemented by many National Research and Education Networks (NRENs) and GÉANT. The general sources of packet reordering are known, but not many realise this, and for many more, reordering is still *Terra Incognita.*

The question of whether packet reordering is pathological network behaviour has been already considered in related work. Following the IP protocol specification, we have to admit that reordering, in the principle, is not a fault of the IP protocol mechanism, but rather something that may happen and we should prepare for it. In other words, the IP protocol provides the delivery of packets, while does not necessarily maintain the order of packets.

In this paper, we show what reordering is, what are the standardisation approaches to reordering metrics, how to measure it and which applications are likely to suffer from it. We also show some experimental results from pan-European packet-reordering measurements.

The definition of packet reordering

Packet reordering is one of the four metrics describing QoS in packet networks: delay, loss, jitter and reordering. Loss, delay and jitter have been well described, their measurement standardised and the influence on user applications quite thoroughly tested. Packet reordering occurs when the order of packets at the destination is different than the order of the same packets at the source. In other words, in a non-reordered packet stream, the packet sequence number of any arriving packet will be lower than the sequence number of the consecutive arriving packet. The following example shows a situation with reordered packets; the reordered sequences are underlined.

source destination

1 2 3 4 5 6 7 8 9 10 ----------> 1 2 4 5 6 3 7 10 9 8

Fig. 1 A sample of a reordered packet flow

The measurements of packet reordering are currently based on the “percentage of reordered packets”. However, this definition, probably derived from the popular “ping-type packet loss” metric, does not provide the precise and adequate information that is required, especially for troubleshooting. Taking the example from Fig. 1, and using a simple “percentage definition” we may state that:

a) there is 30% packet reordering (because packets 3, 9 and 8 are late);

b) there is 50% packet reordering (because packets 4, 5, 6 and 10 and 9 are too early).

Another example shows even worse disinformation resulting from using a percentage definition:

source destination

1 2 3 4 5 6 7 8 9 10 ----------> 2 3 4 5 6 7 8 9 10 1

Fig. 2 A sample of a reordered packet flow

In this case, one may consider the following results:

a) there is 10% packet reordering (because packet 1 is late);

b) there is 90% packet reordering (as all packets except 1 are early)

At the same time, having only a percentage measurement, we do not know other specific facts about the reordering, such as the extent (by how many positions was the packet displaced, how often it happened, was it bursty or flat rate?).

The need for proper packet reordering measurement has become so important due to popularity of TCP protocol and instant measurement of other IP metrics. For example, the TCP protocol can tolerate packet displacement by 1 or 2 positions, and its embedded mechanisms will be able to sort these packets back in order. However, if this level is reached, TCP will still receive the reordered packets (the packet loss at IP level will be 0%) but will drop them, assuming packet loss and adjusting the transmission window (by default, reducing it by half). A similar situation may occur, for instance, when the fact of arrival of a packet with higher sequence number may cause the application to count displaced packets as lost. Similarly, the measurement of reordering can be affected by packet duplicates.

The interaction between IP and higher level protocols has been well described in several works, where we can read that "real protocols and applications are optimised around common case assumptions about how real Internet infrastructures behave under normal conditions. Consequently most protocols assume that corruption, packet loss and reordering are infrequent events or occur primarily under deterministic conditions." It is clear that if these assumptions are not met, the performance of these protocols will suffer.

The fact, that the proper packet reordering measurement is important for particular applications and protocols, has led the Internet Engineering Task Force (IETF) to initiate a packet reordering standardisation track.

The standardisation of metrics and the measurement of packet reordering are currently within the scope IETF IPPM working group, which recently released "draft-ietf-ippm-reordering-10.txt" and "draft-jayasumana-reorder-density-04.txt". Both metrics propose slightly different approaches to reordering.

Reorder Density and Reorder Buffer-occupancy Density (draft-jayasumana-reorder-density-04.txt) are a very simple yet informative metrics for assessing the reordering characteristics and the required reordering recovery mechanisms (there are more metrics defined in the draft, but here we focus only on a selection of them).

Reorder Density shows the distribution of displacements of packets from their original positions, including lost and duplicated packets, within given threshold packet sequence. This means that if the threshold is set to 10, any packet displaced by more than 10 positions will be considered lost.

Reorder Buffer-occupancy Density shows the histogram of the occupancy of a hypothetical buffer, used as a waiting room by early packets (re-ordering buffer). The calculation of this metric is performed upon each packet arrival at the receiver.

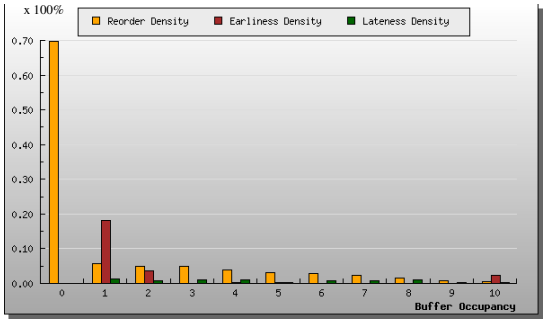

Fig. 3 Sample worst-case Reorder Density graph for the PIONIER-GEANT-CESNET connection

A sample Reorder Density graph is shown in Figure 1. The bars here correspond to the total percentage of packets that were displaced by given number of positions. It is easy to see that some 6% of the packets were late by one position, 20% were early by one position and only 70% of the packets arrived in order. Also some of the packets were displaced by 10 positions.

Other important information that can be derived from the chart is the actual packet loss encountered by the TCP application: in our case, assuming that the TCP protocol can handle the maximum displacement by 2 positions, we have to sum the bars from position 0, 1 and 2. This is roughly 79%, which corresponds to the amount of packets accepted by the TCP protocol. 21% of the packets will be considered lost by TCP

protocol, which will definitively reduce the achievable transmission rate (even though all three networks were highly over-provisioned at a time).

Draft-ietf-ippm-reordering-10.txt defines reordering with a slightly different purpose. Rather than drawing the histogram of the displacement (Reorder Density), the draft defines:

• the Extent of Reordering (showing the displacement for each packet – i.e., how much too early the packet has arrived);

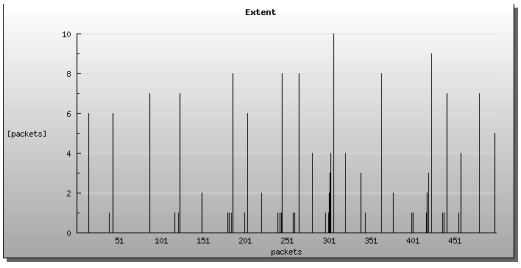

Fig. 4 The Extent of Reordering graph

- the Byte Offset – the storage space in buffer required to restore order;

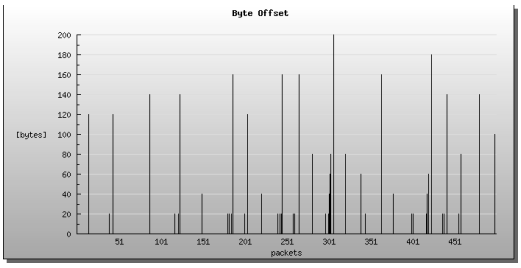

Fig. 5 The Byte Offset graph

- the Time Offset – the amount of time needed to hold the reordered packet until all preceding packets arrive;

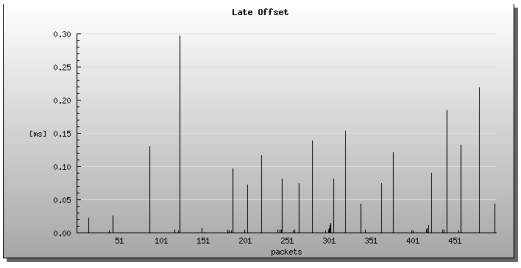

Fig. 6 The Time Offset graph

For each of above metrics, the draft defines the maximum value (i.e. Maximum of Extent), which can be directly used for application tuning. Presented metrics can also be very handy for network troubleshooting, as they allow timely observation of changing network behaviour.

The sources of packet reordering

There are many sources of packet reordering. The most common are:

- all kinds of parallelism in the network;
- network faults;
- improper configuration;
- faulty software
- special QoS/performance configuration.

Probably the most important and common source of packet reordering results from the need to increase the performance of the routing and switching devices in the networks by re-utilising existing solutions in a parallel way, e.g., by adapting the existing equipment to new kinds of interfaces. The reordering here may be introduced by:

Link bundling – in situations where single link capacity is not sufficient between two network devices, link bundling can be applied to increase the total link capacity; the reordering can occur here if the routine scheduling packets to individual physical links works with "packet-based" regime, rather than with "flow-based" one;

Parallel processing within the network devices. The backplane and the packet processors of the network devices have limited capacity. In order to increase the total device capacity, multiple backplane queues and packet processors to interface boards can be implemented in parallel. Here again the extent of reordering depends on the queuing regime. This is often visible in early releases of equipment and less likely to exist in mature solutions, where the processors become more powerful and the parallel solutions are no longer needed.

The root cause for reordering in the above cases is the asynchronous work of the processors and queues of the device. The packet is held in the queue as long as it is necessary to process it, with the time dependent on the packet size, type or additional packet checks resulting from firewall configuration or QoS routines applied. As a consequence, the packets with longer processing time can be bypassed by packets with shorter processing time and which arrived later to other queues.

It is important to note here that the choice of the queuing regime (if possible) cannot be done without penalties. The use of a "flow-based" queuing regime reduces or removes the reordering (all packets from the flow enter the same queue) but this implies that all the packets from a flow have to enter the same queue, usually with the capacity of a fraction of the whole interface. This limits the maximum flow size to the queue size or to be more precise – to the available capacity in the queue selected by the hashing function of the regime.

We will show some practical observations in following paragraphs.

Reordering in Juniper M160, OC-192 interfaces

This is a typical example of packet reordering introduced by the use of four parallel processors, each with the capacity of 2,5 Gbit/s to serve a single 10Gbit/s interface. The independent test results presented at LightReading show that reordering here will not happen until the card has 73% of load, or 56% of load in the worst case, when customer traffic is composed only of 40-byte IP packets.

Unfortunately none of the modern reordering measurement definitions were present at the time of tests, so we are able only to see the "percentage of reordered packets" which is a somewhat imprecise metric.

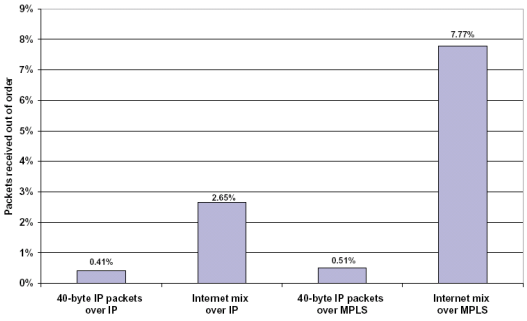

Fig. 7 Juniper reordering measurement results

The charts, however, clearly show that the reordering occurs when different sizes of packets are present, and is rather rare for equal size packets.

The document also states that reordering made by individual routers along the path is not cumulative. This has been confirmed experimentally by our tests in the GÉANT network. We have found that the measured parameters between different GÉANT NRENs do not depend on the number of hops traversed by the packet stream.

Another interesting observation of reordering phenomenon in GÉANT is those resulting from LBE (Less than Best Effort) transmission tests. The experiments here had to investigate the behaviour of LBE traffic in the presence of BE streams. A side observation shows that "the percentage of out-of-order packets is proportional to the packet rate injected; the larger the packet rate, the higher probability of receiving some out of sequence packets". It was further found that reordering here was caused by the platform architecture (M160) and the different weights assigned to Best Effort (BE) and LBE queues. The stated results claim that the proper configuration of weight for different classes can assure no negative effect of reordering on TCP transmission.

Reordering in 10GE cards of Black Diamond switches (BD 6808 and 6816)

Similar to Juniper routers, Black Diamond also implements parallel processing in the 10GE line cards. The problem here lies in the outdated architecture of the switch, which was initially designed to support maximum 1Gbit/s interfaces. Original cards used in BD6806 were supporting 8x1GE interfaces; therefore the backplane connection was also composed of 8x1GE queues. After 10GE standards had been announced, the company released 10GE cards. Unfortunately these cards had to be served by the same 8x1GE queues, already implemented in the backplane. The switch offers two queuing regimes for the 10GE card: packet-based and flow-based. The former schedules each incoming packet to a different queue, introducing significant reordering. The latter preserves packet order, but limits the single flow capacity to the queue size. This is very unfortunate, because in ideal conditions (empty network) the size of a single flow cannot exceed 1Gbit/s (7Gbit/s remains unused and unavailable for that flow). The situation is even worse in the presence of Internet traffic (multiple different flows), where each queue already has some background traffic scheduled. As an example, if the single link has 4Gbit/s of traffic load, it means that average queue load is 500Mbit/s. Each new flow will encounter congestion conditions when its size reaches 500Mbit/s, even if there is still 4 Gbit/s of free bandwidth on the switch. There is no known reordering work-around to solve this problem.

Reordering measurement and results

Packet reordering measurement does not require any sophisticated tools or significant investments. For the purpose of testing the end-to-end reordering in GÉANT, we have used the simple, open source software traffic generator RUDE/CRUDE and some custom-built post-processing scripts. A selection of metrics from both of the mentioned IPPM standards have been implemented and the measurement collected in a mesh scenario, between CESNET, HUNGARNET, HEAnet, LITnet, PIONIER, FCCN and NORDUnet. A special script has been also made available for other users to measure packet reordering in their own samples.

During our experiments we sent test streams with the traffic patterns simulating a few selected applications, including:

- traffic flow designed especially to measure the maximum possible reordering where a burst of small packets immediately follows a burst of large packets;
- bursts of short packets of the same size;
- bursts of long packets of the same size;
- real trace of a JMStudio mpeg video stream (JMStudio is a Java application using JMF;
- 2.0 API to play, capture, transcode, and write media data; JMStudio also uses the JMF RTP APIs to receive and transmit media streams across the network);
- real trace of a VideoLAN Client;
- real trace of MCast6 – our own application streaming in OGG format
- real trace of an IceCast application (OGG)

The following conclusions could be drawn from the above measurements:

a) the streams of packets of the same size were not affected by reordering;

b) the specifics of multimedia streams make them very sensitive to packet reordering;

c) there was not much difference between the results of the "worst case" scenario and multimedia streams;

d) only VideoLAN client was immune to reordering (due to the equal size of the packets).

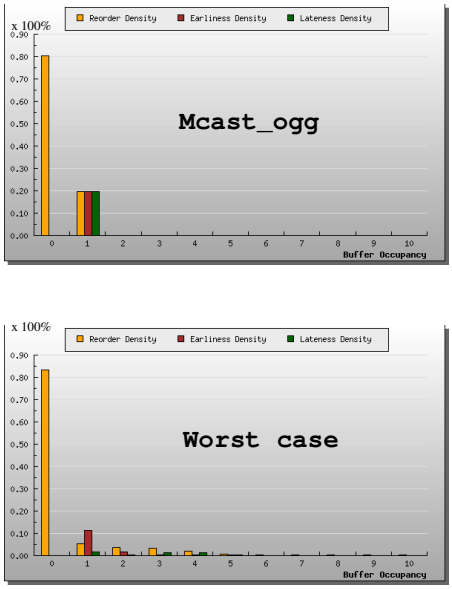

Fig. 8 The results of reordering for multimedia stream and worst case reordering

Implication of reordering on the applications

The tolerance of the application to reordering depends on many factors, including packet transmission rate, packet size, type of the transport protocol used, capacity of receive buffers, application purpose etc.

The most vulnerable applications are those that generate small packets followed closely large packets in a single stream. This implies that slow-rate (in the microscopic scale) applications are not likely to be affected by packet reordering. It has to be noted (and we confirmed that experimentally) that slow applications with large burstiness could be affected, because even though the average transfer rate is low, reordering may occur within packet bursts, where the packets are closer to each other.

Another important issue is the transport protocol. Two transport protocols – TCP and UDP are the most common in Internet. TCP uses a transmission window mechanism, which adapts to current network conditions (packet loss) by resizing itself. However, in many cases, TCP will not distinguish packet reordering from packet loss, because it will treat a packet reordered by three positions (or a packet which came three packets later than expected) as lost packet. The detection of false packet loss will cause the transmission window to downsize, affecting overall transmission throughput. Related work shows some approaches to more reordering-resilient TCP implementations. During our

experiments, we have observed reordering greatly exceeding the order of three packets over many European links, which will definitely influence TCP throughput of many high-bandwidth, demanding applications.

The UDP protocol is mainly used for media streaming over the Internet and for highly interactive services, such as videoconferencing or Voice over IP. These services have one important feature – if the data (such as digitised speech during videoconference) is not delivered on time, it can be as well discarded, because after certain time period, this data is no longer needed. During our experiments we have tested the traffic patterns of various applications, including JMStudio, Video Lan Client and IceCast Ogg audio streams to asses their vulnerability to packet reordering.

One of the results of traffic analysis is that some application modifications are necessary to deal better with reordering-impaired networks. As an example, the trace from IceCast Ogg shows the following packet pattern (for two consecutive voice packets):

frame size	**time from the last frame**	
1024	0.349454	
1024	0.000089	
1024	0.000086	**(1st voice packet)**
1024	0.000097	
88	0.000004	
1024	0.319239	
1024	0.000087	
1024	0.000087	**(2nd voice packet)**
1024	0.000087	
118	0.000015	

We can see from this pattern that when the application sends the traffic in a bursty way, one voice packet is packed into 5 frames, with the last smaller one, carrying the remaining part of data. The frames carrying one packet are sent almost simultaneously, and then the application waits to assemble another voice packet. This application has been found to be vulnerable to reordering, as the last small packet usually bypassed one preceding packet during all transmissions. However, the application uses the TCP protocol for transmission, allowing compensation for such a level of reordering.

Shall we worry about packet reordering?

The answer to that question is not straightforward. Reordering is quite common in Internet now and will stay common in the future. The only way to harness reordering is to learn what it is, what are its implications for applications and how to protect them from this phenomenon. However, the biggest problem is that the most common protocols, invented in the time of kilobit/s transmissions, are still the main transport protocols of modern gigabit networks. They cannot cope well with reordering, so one option would be to look towards limiting the reordering in the network devices. The practice shows that even the largest and strongest market players cannot avoid troubles with reordering in their devices. It is very important, then, to recognise all the pitfalls related to reordering in network equipment and to properly evaluate any device to be implemented in specialised broadband networks. This is especially important for the research community, requesting undisturbed transmission of high-bandwidth streams. At the same time, more work is required to assess the influence of reordering on various transport protocols and to develop new kinds of reordering-resilient transport protocols.

Acknowledgements

The authors would like to thank our European partners who kindly granted us the access to the test machines – CESNET, FCCN, HEAnet, HUNGARNET, LITNET and NORDUnet.

References

1. http://www.ietf.org/internet-drafts/draft-jayasumana-reorder-density-04.txt
2. http://www.ietf.org/internet-drafts/draft-ietf-ippm-reordering-10.txt
3. J. Bennett, C. Partridge, and N. Shectman. Packet reordering is not pathological network behavior. IEEE/ACM Transactions Networking, 7(6):789-798, Dec. 1999.
4. B. Chen and R. Morris. Flexible control of parallelism in a multiprocessor PC router. Proceedings of the 2001 USENIX Annual Technical Conference, pages 333-346, June 2001.
5. http://rude.sourceforge.net/
6. M. Zhang, B. Karp, S. Floyd, L. Peterson, RR-TCP: A Reordering-Robust TCP with DSACK
7. http://java.sun.com/products/java-media/jmf/2.1.1/jmstudio/jmstudio.html
8. http://www.videolan.org/vlc/

9. http://www.icecast.org/
10. http://www.cnaf.infn.it/~ferrari/tfngn/lbe/results/lbe-geant/
11. http://herodes.redes.upv.es/rap/Routers%20Altas%20Prestaciones/core_router_test.pdf
12. John Bellardo, Stefan Savage, „Measuring Packet Reordering"

Authors biographies

Bartosz Belter received the M.Sc. degree in Computing Science from Poznan University of Technology in 2002. He works in Poznan Supercomputing and Networking Center in Network Research and Development Department as a Network Application Developer. His main IT research interests are software engineering, Java language and network technologies.

Artur Binczewski received the M.Sc. degree in Computing Science from Poznan University of Technology in 1993. His research interests concern computer networks, routing, multicasting and management. He is the Manager of Network Department at Poznan Supercomputing and Networking Center.

Michał Przybylski received his M.Sc. degree in Computing Science from Poznan University of Technology in 2002. He has been working in Poznan Supercomputing and Networking Center since 1999, he currently leads Network Research and Development Group. His research interests include broadband, optical networking and new network services.