Audio Engineering Society
Convention Paper
Presented at the 124th Convention
2008 May 17–20 Amsterdam, The Netherlands

"Audible ICMP Echo Responses for Monitoring Ultra Low Delayed Audio Streams"

Alexander Carôt[1], Alain B. Renaud[2] and Christian Werner[1]

[1]*Institute of Telematics, University Lübeck, Germany*

[2]*Sonic Arts Research Centre (SARC), Queen's University Belfast,Northern Ireland*

Correspondence should be addressed to Alexander Carôt (`Alexander@Carot.de`)

ABSTRACT
Playing live music on the Internet is very demanding in terms of delay, loss or jitter and hence requires extremely reliable network conditions. Jitter is the most problematic factor because it has a direct influence on the required network buffer sizes for receiving low delay audio streams. Therefore measuring the amount of jitter is a very complex task due to the multi-hop architecture of the Internet. So far it has been impossible to know at which hop these delay variances appear. The authors propose a solution that is able to generate an audible impression of the jitter problem for each hop.

1. INTRODUCTION

In order to measure the actual delay between two hosts on the Internet the most common approach is the use of Echo request and response messages provided by the *Internet Control Message Protocol (ICMP)* [9]. The most popular implementation is the *ping* command, which is part of most modern operating systems. By default the ping command sends a sequence of echo request messages to the remote destination. It usually includes the sending time as part of the payload. At the destination the ICMP implementation recognizes this specific type of packet and echoes it back to the sender. Now, the included time stamp gets subtracted from the current system time in order to calculate the roundtrip time (RTT). By measuring the RTT several times an average value can be calculated. The derivation of this average value is called jitter [12].

In order to analyze the route between the sender and the destination host there is a second commonly available monitoring tool called *traceroute*. It makes use of the *Time To Live (TTL)* header field of the IP header [8]. The numeric value of the TTL field is decremented by (at least) one if the packet is forwarded to the next hop. If a value of zero is reached, the packet must be discarded. This mechanism prevents packets from circulating endlessly on the Internet in case of a faulty routing setup.

The traceroute command uses the TTL value to reveal the routing hosts on the way between source and destination. It starts sending out an arbitrary UDP packet [7] with a TTL value of one. The first hop decrements it to zero, drops the packet and sends back an *ICMP Time Exceeded* message. Hence, the source host can detect the first hop by reading the IP sender address of the ICMP message. Next, the traceroute command repeats this process with a stepwise increased TTL value until all hops on the route are resolved.

By combining ping and traceroute the values of RTT and jitter can be measured for any hop of the route.

2. PROBLEM

Since the amount of jitter significantly depends on the actual network load [5] it is mandatory to achieve realistic measurement conditions. Hence, in case of a network music performance, using ping and traceroute without the actual audio payload does not represent a realistic network scenario.

One obvious solution would be to establish the audio transmission and perform the jitter measurements in parallel. A severe drawback to this approach is that the receiver must be up and running. In the case of live music performance this typically implies that the remote musician must start according services. With additional network components, such as NATs or firewalls, the setup becomes even more complicated and inconvenient to use [3].

Besides this, the measured jitter is only represented as a set of numeric values. These do not necessarily correspond to the perceived quality of the transmitted audio streams, because it depends on several potentially undetermined technical parameters such as packet and buffer sizes.

3. APPROACH

A significant improvement would be to include the audio information in the packets for jitter measurement. These would include the actual audio data combined with a time stamp – analogously to ICMP Echo Request messages.

This approach would require to place a UDP echo service on every hop of the route in order to mirror back the relevant audio stream. This in turn would give the listener an audible impression of the current jitter conditions on this specific part of the route. However, it is usually unpractical to set up new services on foreign Internet routers.

In that context we examined whether an ICMP implementation, which is present at any Internet router, could behave in the same way as a UDP echo service does. This would enable the user to test any hop on a given route without setting up a custom endpoint. In fact, both UDP and ICMP are connectionless protocols which have so far been used for completely different purposes.

This paper demonstrates that it is possible to include audio payload data in ICMP Request messages and use Echo Responses as an audible representation of the current jitter conditions on the route to an arbitrary Internet host. As we will demonstrate, this is a very convenient and flexible method not only for jitter measurements but also for a great variety of other audio relevant network parameters. The most important practical benefit in comparison to existing solutions is the possibility to measure without setting up custom endpoint services on the remote side.

4. REALIZATION

So far, most tests and measurements for network music performance were typically realized with broadband Internet connections [5] such as Internet2 [2] or GEANT [1], where jitter typically occurs in a very low dimension [11] – in contrast to DSL networks with lower bandwidths, where delay variations with higher values are more likely. Hence we consciously chose one DSL endpoint for our experiments, in order to examine whether the ICMP protocol behaves in the same way as a custom UDP mirror does. We set up two test peers – one on a home DSL connection in Luebeck/Germany (16 Mbps downstream / 700 Kbps upstream), a second at SARC (Sonic Arts Research Center) in Belfast/Ireland (Gigabit backbone connection). Both peers were equipped with the Soundjack software [4] which is

Samplerate	48 kHz
Framesize	256 Samples
Decimation Factor	8
Soundcard Periods	2
Bitdepth	16 Bit
Audio Channels	1
Network Buffersize	1
Audio Blocking Delay	5,2 ms
Audio Packet Size	64 Bytes

Table 1: Soundjack audio settings

able to generate low delayed audio stream as UDP packets and ICMP packets at any desired latency and bandwidth. With regard to realistic parameter settings [6] for network music performances, Soundjack was configured with the settings outlined on table 1.

These settings normally result in a fixed audio packet size of 512 bytes but due to a decimation factor of 8 the block size results in 64 bytes in order to fulfill the DSL upload restrictions [3]. These audio blocks are sent every 5.2 ms for the sending side and due to only one network buffer, Soundjack parses any received audio buffer directly into the soundcard on the receiving end [3]. This small network buffer doesn't introduce additional delay and would automatically indicate delay variances in the network stream through the corresponding audio dropouts. Since special emphasis on this effect is desired, we consciously avoided larger jitter buffers which are typically applied in Internet telephony, video conferencing or generally in realtime Internet traffic [10].

Based on these settings, one UDP and one ICMP stream were sent in parallel to the Belfast machine, which reflected both of them back to the Luebeck machine. The reflection of the ICMP packets was achieved by using an echo request which mirrors an incoming packet back to its sender. For the UDP stream a custom UDP mirror service was integrated into the Soundjack software. Once the streams arrived back at the sending host, the dropouts resulting from network jitter or packet loss were measured. For packets which arrived within the required time boundaries led to a reliable audio playback, a "0" was written into a file. When an audio underrun was discovered, a "1" was written into the file. The file was later used as the source for the generation a dropout graph. One minute of streaming audio was generated, which equaled to a number of about 11000 audio packets. The steam was sent, reflected, received back and measured with the dropout results for each packet shown in the following graphs. Whenever a dropout appeared the graph shows a black line. Due to the relatively large number of packets a single dropout line appears quite thin and the line gets darker as the number of dropouts augments.

The aim of the first measurement was to find out how far an ICMP audio stream corresponds to a reference UDP audio stream in terms of network in general and especially in terms of network jitter. The measurement was performed on Sunday March 9th 2008 at 2:00 am to achieve a low number of dropouts for better visualization and comparison.

The graph in figure 1 shows that the behavior of the ICMP stream almost equals the one of the UDP stream. There is a slight variation in the amount of dropouts but the times of dropouts are precisely the same. As an exception to this rule, the UDP response graph shows an additional dropout line at the second position from the left. We concluded that this small inconsistency was caused by slight variances of the network load situation. Physically the sending events of the UDP and ICMP packets are not exactly simultaneous but happening in a sequential order. Therefore, slight variances are plausible. Despite these slight inconsistencies, the results show clearly that the ICMP behavior closely relates to the UDP dropouts.

Hence, in order to measure potential sources of dropouts the ICMP echo request could be reflected by any router on the way to the Belfast destination. As a first step we resolved each host by a traceroute, which indicated the hops on the route shown in table 2.

As already mentioned, the existence of a host on the Internet is commonly checked with ICMP echo responses, typically implemented by the ping command. However, in terms of special network security, network administrators sometimes block ICMP echo requests on some hosts. If this is the case, neither an ICMP request or audio stream would function in the described way. Therefore, as a second step we had

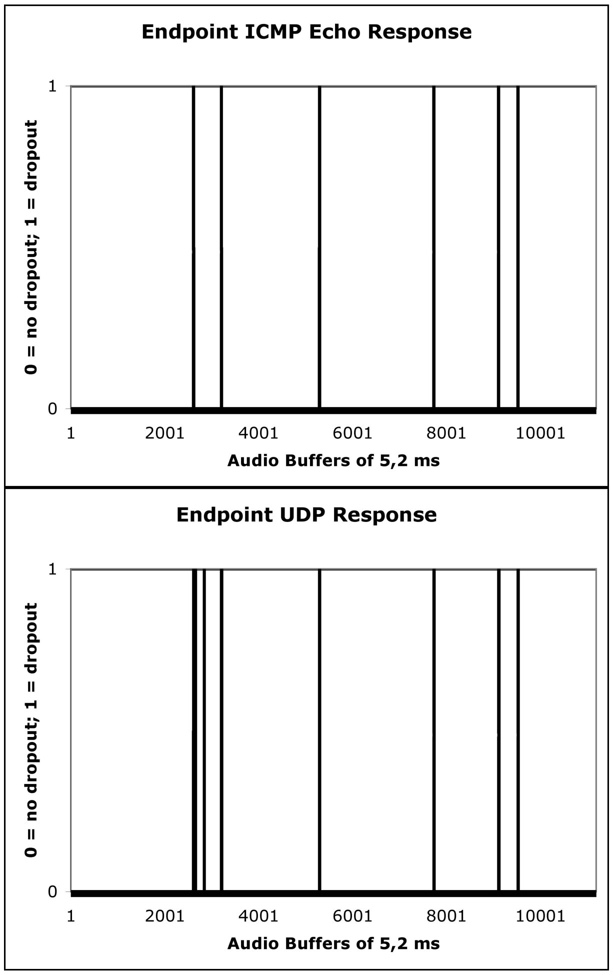

Fig. 1: Similarity between ICMP and UDP endpoint monitoring

to find which of the involved hops had ICMP echo response enabled . By applying the ping command, it turned out that except for the hop numbers 11,12 and 13 all other hops had ICMP echo responses enabled.

The third step involved measuring the hops with the same one minute streaming test as with ICMP data in parallel to the endpoint in Belfast. Hop 14 was used as the reference UDP mirror stream. We started with the closest hop, the DSL router itself, our testmachine was connected to. Due to no additional cross traffic on the router, figure 2 shows that between the cable linked endpoint and the DSL router no dropouts appeared. At the same time, the UDP reference stream showed three dropouts caused by jitter on further involved hops.

hop number	hop IP address
1	89.246.3.71
2	213.30.195.181
3	213.248.77.105
4	80.91.251.81
5	80.91.254.219
6	80.91.252.10
7	213.248.104.154
8	146.97.33.17
9	146.97.42.182
10	195.194.169.250
11	143.117.0.9
12	143.117.254.33
13	143.117.255.162
14	143.117.78.184

Table 2: Hops between Luebeck and Belfast

Afterwards we measured hop 2 to 10 respectively. Even though most of the ICMP echo responses were received as expected, in certain intervals for the majority of hops we experienced some further inconsistencies regarding the dropout behavior. These inconsistencies resulted in strong bursty jitter and additional dropouts for the ICMP stream. We could observe this behavior even on hop 2, which is illustrated in figure 3. This was apparently caused by the ICMP implementation on the reflecting router, which obviously had problems responding to larger amounts of ICMP echo requests.

For hops 6 and 9 the resulting graphs in figure 4 and figure 5 show the expected and desired behavior. The UDP reference stream indicates dropouts at the same time with similar values. Additionally we can observe that the results in figure 5 indicate more dropouts for the ICMP stream than in figure 4 due to a larger hop count.

5. CONCLUSION

Though the ICMP protocol is mainly intended for network measurement and normally delivers technical information such as delays and error messages in a numerical representation, it can also be used

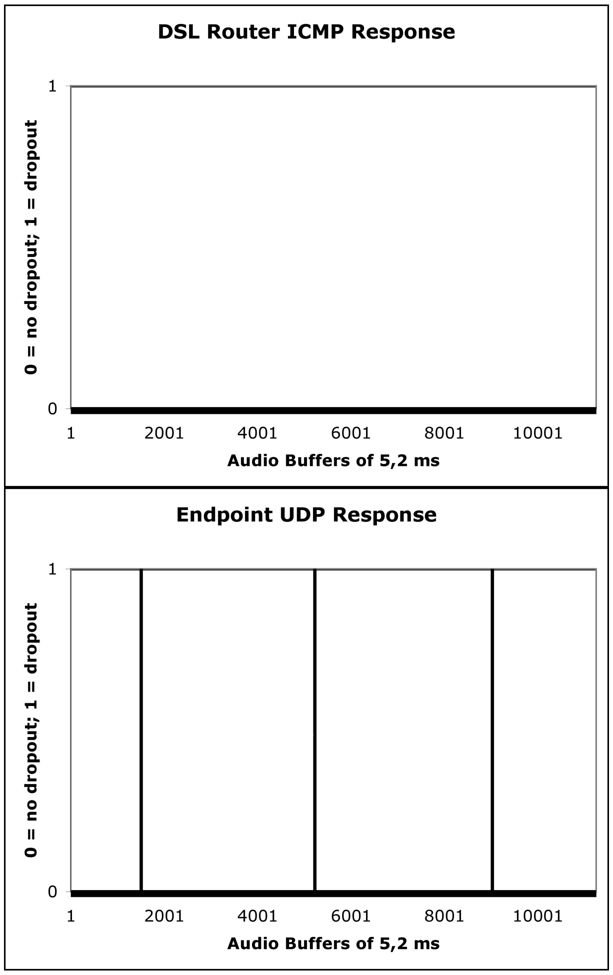

Fig. 2: ICMP monitoring of hop 1 (DSL Router) in comparison with UDP endpoint monitoring. No dropouts occur on hop 1.

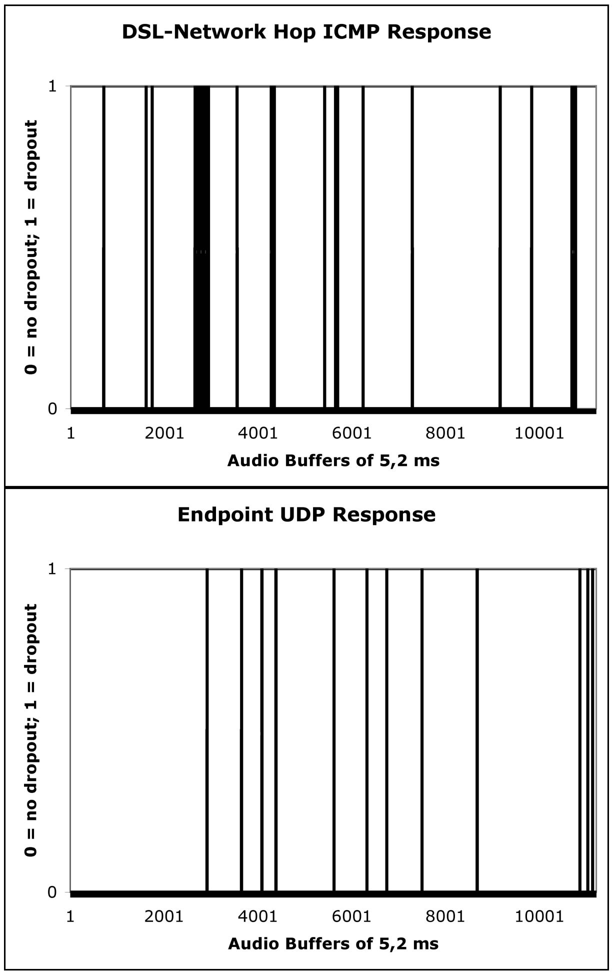

Fig. 3: Inconsistent measurement results for hop 2

for the transportation of audio data. The direct comparison shows, that in terms of packet delivery time and dropout behavior in our example setup, an ICMP stream behaves almost precisely in the same way as a UDP stream does. However, in theory ICMP traffic might be routed completely differently than UDP traffic. This practical experience makes us believe that our approach works in most of all test cases.

In terms of low delay audio streams, network jitter represents the most significant problem. It can casually appear on any involved hop but unfortunately it has so far been impossible to determine the actual location. The principle of audible ICMP echo responses now offers the possibility to use such a mechanism to reflect audio streams and in turn retrieve information about the current jitter for the specific network segment. In that context it became clearly audible and visible that in case a delay variance occurs on a specific hop, it appears on the endpoint hop as well. Though jitter is a casual phenomenon, which only appears with a certain probability depending on the actual network conditions, it became clearly obvious that the jitter increases with the number of hops. Each hop represents a network segment with its own probability for network jitter,

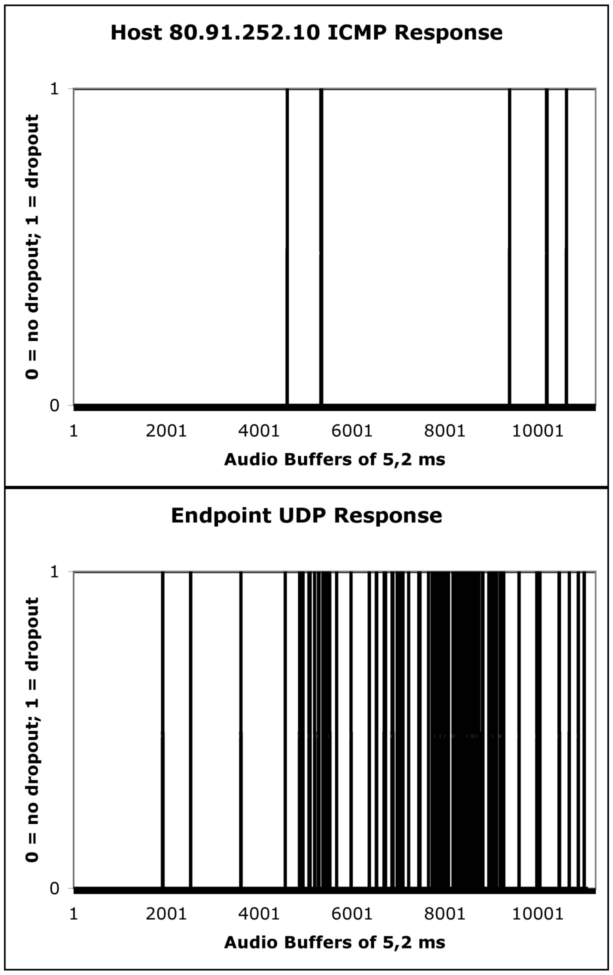

Fig. 4: Consistent measurement results of hop 6

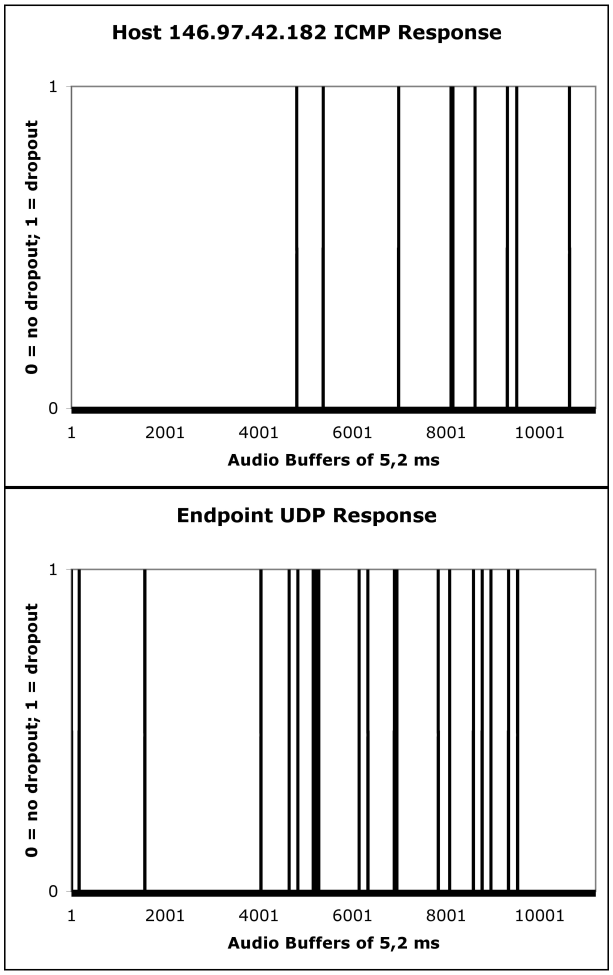

Fig. 5: Consistent measurement results of hop 9

so that the sum of probabilities results in the endpoint jitter probability.

However, in some cases, the measurements are coherent to what we stated in this paper. Firstly the values of dropouts differ significantly. Secondly dropouts appear in the ICMP stream which do not exist in the UDP stream. These exceptions can be explained by the fact, that each stream packet travels the route twice – to the destination and back – and since the network situation might have changed slightly after a packet has reached the reflector, different values can be expected. Due to this effect, there is no need to expect a total correlation of both ICMP and UDP streams, in order to prove the efficiency of ICMP audio streams.

We believe that audible ICMP echo responses are a very practical and convenient approach for monitoring and debugging specific network segments in the context of network music performance.

6. FUTURE WORK

Though we could consider the idea of audible echo responses as a useful feature, their precise examination requires further investigation. In the future the Soundjack software will be equipped with the option of sending numerous ICMP streams instead of just one. This way, all hops will be tested at the same time to be compared instantly within the same measurement, rather than in a successive manner, where the probability of audio dropouts might influence the test results. Since numerous audio streams require a higher amount of bandwidth, we will use a symetric broadband connection instead of an upload restricted DSL link. These future experiments will allow us to find a more precise answer on why some hops temporarily malfunction when loaded with ICMP streams. In that context we will further investigate the various sources of network jitter and will use audible ICMP echo responses as the main measurement tool.

7. REFERENCES

[1] Geant website, March 2008. `http://www.geant.net`.

[2] Internet2 website, March 2008. `http://www.internet2.edu`.

[3] Alexander Carôt, Ulrich Krämer, and Gerald Schuller. Network music performance (nmp) in narrow band networks. In *120th AES convention*, Paris, France, May 2006.

[4] Alexander Carôt, Alain Renaud, and Bruno Verbrugghe. Network music performance (nmp) with soundjack. In *6th NIME Conference*, Paris,France, June 2006.

[5] C. Chris, S. Wilson, R. Leistikow, D. Chisholm, and G. Scavone. A simplified approach to high quality music and sound over ip. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00)*, Verona,Italy, December 2000.

[6] Ken C. Pohlmann. *Principles of Digital Audio*. The Mcgraw-Hill Companies, fifth edition edition, 2005.

[7] J. Postel. RFC 768: User datagram protocol, August 1980.

[8] J. Postel. RFC 791: Internet Protocol, September 1981.

[9] J. Postel. RFC 792: Internet Control Message Protocol, September 1981.

[10] R. Reynolds and A. Rix. Quality voip – an engineering challenge. In *BT Technology Journal*, November 2004.

[11] R. Sinha, C. Papadopoulos, and C. Kyriakakis. Loss concealment for multi-channel streaming audio. In *Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Monterey, California, USA, June 2003.

[12] Andrew S. Tanenbaum. *Computer Networks*. Pearson Studium, fourth edition, 2003.