



HAL
open science

Non-Neutrality Detection of Video Streaming: Wehe Tool Demonstrator and Efficiency

Antoine Lesieur, Patrick Maillé, Bruno Tuffin

► **To cite this version:**

Antoine Lesieur, Patrick Maillé, Bruno Tuffin. Non-Neutrality Detection of Video Streaming: Wehe Tool Demonstrator and Efficiency. VALUETOOLS 2024 - 17th EAI International Conference on Performance Evaluation Methodologies and Tools, Dec 2024, Milan (Italie), Italy. pp.1-4. hal-04222971

HAL Id: hal-04222971

<https://inria.hal.science/hal-04222971v1>

Submitted on 29 Sep 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Non-Neutrality Detection of Video Streaming: Wehe Tool Demonstrator and Efficiency

Antoine Lesieur
Inria, Univ. Rennes, CNRS, IRISA IMT Atlantique, IRISA, UMR CNRS 6074 Inria, Univ. Rennes, CNRS, IRISA
Rennes, France
antoine.lesieur@inria.fr

Patrick Maillé
Rennes, France
patrick.maille@imt.fr

Bruno Tuffin
Rennes, France
bruno.tuffin@inria.fr

Abstract—With network neutrality regulation imposed worldwide at different extents, there is a need for monitoring tools to verify if Internet service providers (among other actors) comply with the existing rules. Among the very few maintained tools, Wehe has been developed by Northeastern University and has been publicized by the French regulator, ARCEP. The tool runs traces and produces a positive or negative response about service differentiation, but users have to trust that result without any visual perception of the degradation. Our goal in this paper is to describe a demonstrator of Wehe for video streaming, from which we can simultaneously see a baseline video and its manually-differentiated counterpart, as well as run Wehe to evaluate its response with respect to the perceived quality degradation. The demonstrator allows to study the efficiency of Wehe in order to propose potential improvements, if needed.

Index Terms—Network neutrality, Monitoring tool, Video streaming.

I. INTRODUCTION

Since the late 90s, *network neutrality* has been the subject of a sensitive and lasting debate. Basically, network neutrality means that Internet Protocol (IP) packets are treated equally within the network, regardless of the type of content, origin or destination, and type of platform, application or device they are related with. This notion was defined after actions from Internet Service Providers (ISPs) who were wishing to differentiate service and/or to charge (distant) content providers for the traffic going through their network to earn extra revenue and fund their infrastructure [1]–[3]. That move provoked strong reactions from user associations and Content Providers (CPs) who were rather wanting to keep an *open* and neutral network. Different regulations have been passed throughout the world based on different interpretations about what should be considered a neutral behavior [3]. We even observed changes of mind such as with the 2017 neutrality repeal under the Trump administration in the US. Because of this, and due to evolving technologies and new issues popping up, the debate is often seen as never ending. As illustrative and important examples entering the debate, we can cite sponsored data, or slicing in 5G networks [3].

But as soon as rules are enforced, there is a need for measurement tools to ensure that the telecommunication actors comply with defined neutrality principles, or just to monitor activities to point out potentially harming behaviors. Actually few tools have been developed, mostly designed

by the academic community, but often restricted in terms of infringement detection, and rarely maintained; see [4], [5] for recent surveys. Though, a tool called Wehe may be seen as standing out, jointly designed by Northeastern University and the French regulator ARCEP [6]. This tool is maintained and distributed on Apple and Android application stores [7] and its code is open source¹. The tool is analyzing potential traffic throttling on the Internet. Due to its diffusion and official role, we focus on it in this paper.

We carried out a first analysis of Wehe in [8], investigating the playroom left for ISPs to differentiate traffic before being detected: By computing the detection probabilities and estimating the potential benefit of a revenue-interested operator, we compared the main differentiation types (throughput, packet loss and delay) that an operator could implement.

The present paper has a different goal. Wehe runs like a black box, by basically launching two similar traces: a baseline corresponding to a traffic an ISP can identify and differentiate, and an encrypted one which should be treated neutrally since impossible to identify. An analysis of the difference between the outputs of the two traces is performed by the tool, but unfortunately no output traffic is “visualized” by the end user, who has to trust the simple (positive, negative or inconclusive) answer provided by the tool. We propose here to present a demonstrator of Wehe on video streaming, showing simultaneously a baseline video and the same video degraded (the level of degradation being tuned by the user thanks to buttons) and running Wehe for the selected level of differentiation. The goal is to relate the *observed* degradation and the potential differentiation detection. It allows to evaluate the practical level of efficiency of Wehe.

The remaining of the paper is organized as follows. Section II briefly describes how Wehe works, then Section III presents the demonstrator and provides links to download it and/or to observe how it operates. The analysis of the detection algorithm for video streaming is presented in Section IV. Finally we conclude in Section V and give direction of future research.

II. WEHE

Wehe is an active measurement tool targeting mainly wireless networks and mobile terminals, on which most neu-

¹See <https://dd.meddle.mobi/codeanddata.html>

trality infringements have been observed. Thanks to an app available from Android or Apple application stores, the tool tests traffic usually subject to differentiation (such as, among others, Hangouts, Netflix, Skype, Spotify, Viber, Amazon and Youtube). Basically, it replays twice a prerecorded traffic between the client and a Wehe server. The first replay is the original one while the second one is encrypted so that an ISP cannot identify it, and therefore not discriminate it. In both cases, the replayed traffic has the same shape, the same packet sizes, protocol headers and inter-packet times, the only difference being the encryption or not of the payload, so that the outputs should be the same if no differentiation is performed. Statistical tests are then performed to compare the throughputs of the output traffic by determining if a difference is significant: this would mean a discrimination between the identified and encrypted traffic. More details can be found in [6] and Wehe python source code can be found at <https://github.com/NEU-SNS/wehe-server>.

The app is practical and easy to launch: the user selects as an input which traffic should be tested and after a few seconds a positive, negative or inconclusive response on non-neutrality is provided. But it works like a black box: users do not have any visual perception on the level of difference between the two traces: they have to trust the tool without any view on how efficient it is.

III. A DEMONSTRATOR

In order to study/illustrate the efficiency of Wehe, we have designed a demonstrator coded also in python and which can be freely downloaded from

https://files.inria.fr/wehe_demonstrator/package_wehe.zip

It contains a “Readme” file giving installation instructions.

The demonstrator generates a topology made of three virtual machines. One is acting as the Wehe server and one is acting as the client device (or end user). Communications between client and server are done thanks to the third virtual machine called the middlebox on which is run a Linux Traffic Control utility with a netem queueing discipline [9] for classifying and differentiating traffic. Netem allows to limit or deteriorate three performance characteristics, potentially combined: throughput, packet loss rate and delay applied to classified packets.

In order to visualize the correspondence between a chosen level of differentiation and Wehe detection, we consider the impact on video streaming, since it is the type of traffic of most applications targeted by the Wehe app. The demonstrator simultaneously displays two screens: one for the nominal video (not discriminated, corresponding to the encrypted one), and one for the potentially differentiated, with discrimination levels on throughput packet loss or delay selected by the user thanks to buttons. This is illustrated in Figure 1.

The videos are played over HTTP using adaptive bitrate (ABR) where the content is segmented and encoded at multiple bit rates, and the segment displayed corresponds to the available

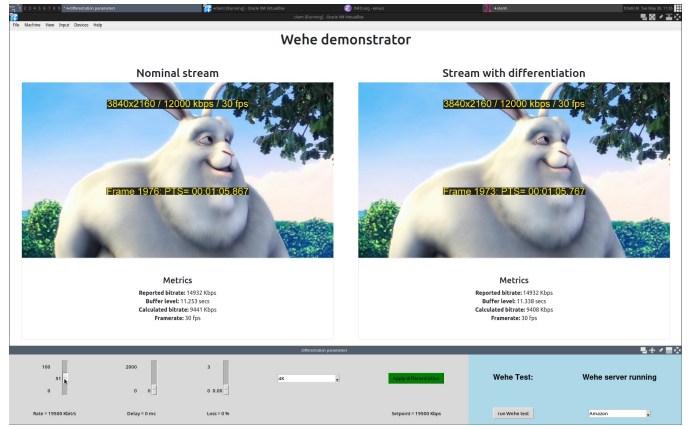


Fig. 1. Components of Wehe demonstrator: screens of the differentiated and non-differentiated videos, box for discrimination parameters, and corresponding Wehe test.

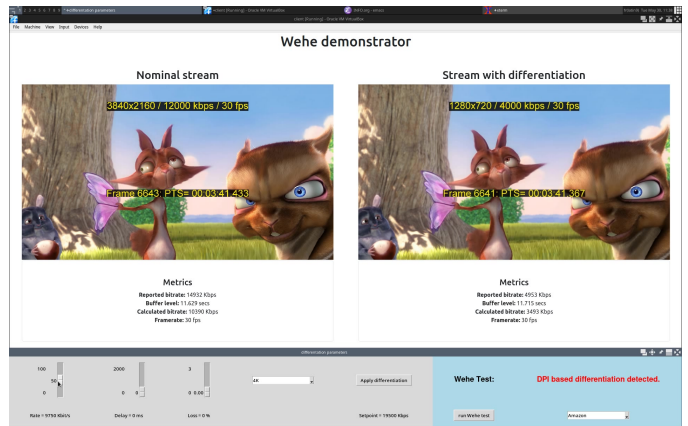


Fig. 2. Wehe test, with positive detection.

throughput. To facilitate the analysis, the demonstrator displays on the videos their resolution and the actual throughputs. When the differentiation parameters are selected, the user can directly see the impact on the video quality. A Wehe test can then be performed by selecting a trace on the bottom right part of Figure 1, for which a (similarly) differentiated and nominal traffic are sent and compared and the result (differentiation detected, not detected, or inconclusive test) of the test presented.

As illustrative examples, a positive test is presented in Figure 2 and a negative one in Figure 3 (despite a reduced resolution with the selected parameters for this last example).

For those not wanting to install the demonstrator but still interested in getting a more detailed idea about how it works, a short demonstration video is available at

<https://youtu.be/SgB4gIBvX8w>

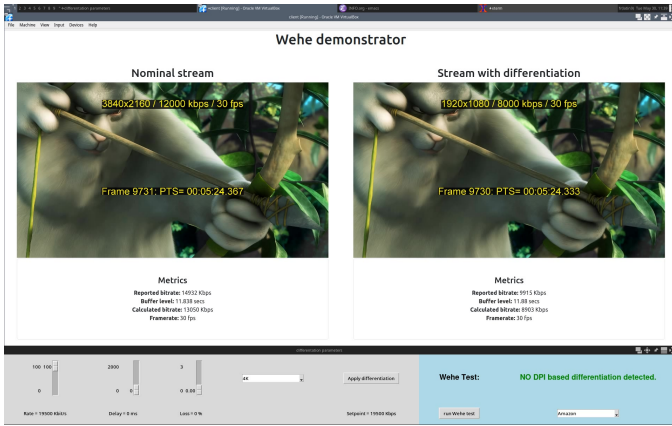


Fig. 3. Wehe test without detection.

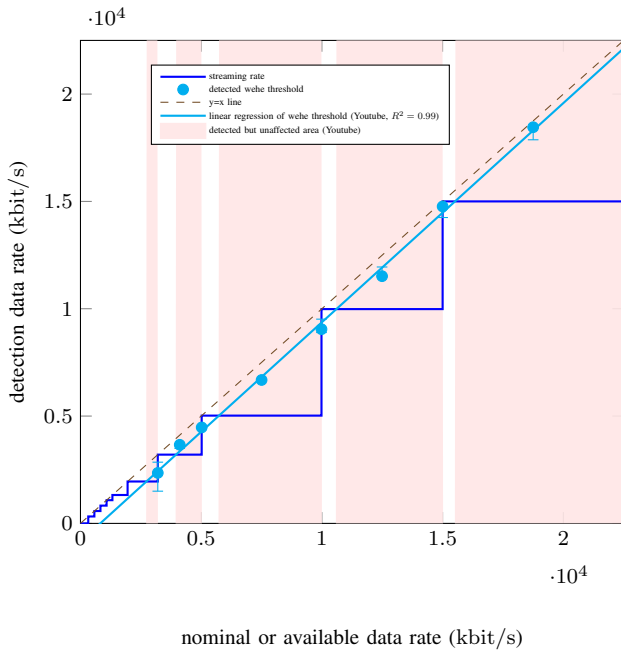


Fig. 4. Detection threshold of Wehe (Youtube trace) and video streaming.

IV. EFFICIENCY RESULTS

This section relates the Wehe detection threshold with the ABR resolution degradation. To do this, Figure 4 displays the Wehe detection threshold and ABR resolution degradation in terms of the actual available data rate when the differentiation is done by reducing the sending rate. More precisely, the x -axis is for the available throughput, and the points give the median value of threshold, over ten independent sets of experiments, of the (reduced) throughput under which differentiation is detected. For each set of experiments the threshold is determined by dichotomy, as follows: we start with a search interval between half of- and the nominal throughput (assuming reducing the throughput to its half would mean detection), and reduce that search interval by performing a new experiment with its middle value as the degraded

throughput, stopping when the length of the interval is below a given relative value of the nominal throughput; we then consider the last found point as the threshold of the set of experiments. The intervals around the points represents the interdecile ranges (that is, the difference between the first and the ninth deciles over the ten experiments). Detection is indeed a random variable since there is some randomness when traces are played,

A linear regression curve (with coefficient of correlation close to 1) on the detection threshold with respect to the available throughput is computed, and shown in Figure 4. The curve $y = x$, that would correspond to perfect detection, is displayed to better grasp the percentage of degradation for which a detection is observed: the vertical difference with the diagonal gives the level of throughput degradation an ISP can operate without being detected. The regression and $y = x$ curves are close to parallel, with a leading coefficient 1.02 for the linear regression meaning that an *absolute* difference of throughput is detected while we could have expected to detect a relative one. Basically with the YouTube trace presented here, a difference of 800 kbit/s is the upper limit before being detected. The blue curve shows the actual streaming rate of ABR, representing the rate of the segment observed by the streaming tool for the nominal video. It is of course a stair-step function of the available data rate, due the discrete number of resolutions. To see the effect of differentiation on the actual streaming rate, the streaming rate of the differentiated video (at the threshold value) is “just” the blue curve translated by a value of 800 kbit/s. The colored sections indicate the domain where degradation is detected by Wehe and actually does not result in a loss in video resolution. This shows that for most values of data rate, the streaming (ABR) rate of the differentiated (at the threshold level) and nominal videos are the same, so visually no differentiation is observed but Wehe still detects it, suggesting the tool is sensitive enough to prevent ISPs from implementing harmful differentiation. There are short intervals though, when the streaming segment rates reach the data rate, where we do not have a detection but the streaming rate is different. They are described in Table I.

TABLE I
INTERVALS WHERE DETECTION AND ABR SWITCH OVERLAP

overlap intervals (kbit/s)
0-2720
3208-3941
5000-5717
10000-10579
15000-15507

Note that our previous analysis is based on running Wehe on a YouTube trace, but several exist and we need to test if detection is trace-sensitive. Figure 5 shows how different the detection threshold is compared to the different traces (Here, YouTube, Amazon and Netflix). Hence, the shape of the trace, its length, its sequenciation, etc. have an impact on the efficiency of the differentiation detection. For instance, the detection threshold with the Amazon trace is much lower

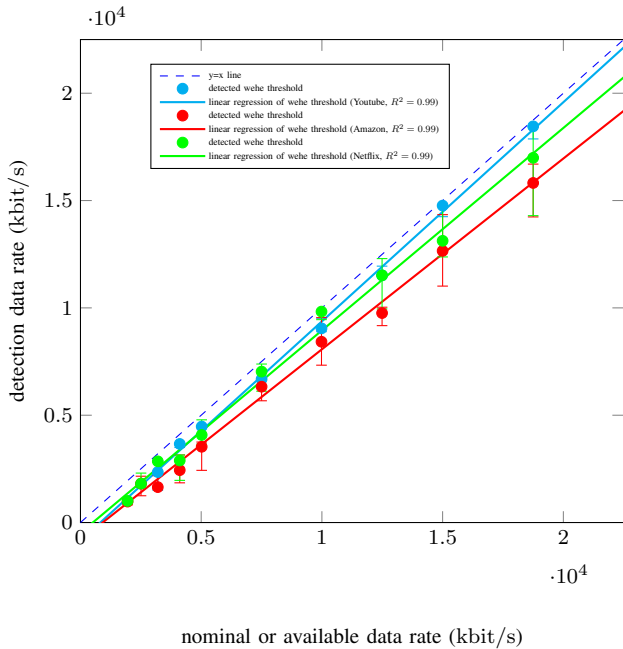


Fig. 5. Variation of the detection threshold for different traces.

than with the Youtube trace (2.5 Mbit/s against 800 kbit/s at a rate of 15 Mbit/s). This means that Wehe is much more sensitive to a differentiation applied to the YouTube trace than to the Amazon one. But would it be the same if we take other YouTube and Amazon traces? Is it something worth being investigated in the future even if the impact is more quantitative than qualitative. The threshold is therefore trace-dependent; considering several traces for the same source/application would be worthwhile to investigate if the same type of gaps are observed for the same type of traffic.

V. DISCUSSION AND CONCLUSIONS

Our work allows to analyze the efficiency of the Wehe non-neutrality detection tool on streaming applications, which are the applications the most tested by the tool. We are able to relate the differentiation detection event to the observed streaming rate, something not possible with the application, which just had to be trusted.

In summary, Wehe seems to work fine for a large part of the data rate values (with a possible differentiation detection even if no difference in the streaming rate is observed), while for a small range of values close to the jumps in streaming rate, Wehe does not detect a differentiation even if the streaming rates of the two videos are different. In addition, since the traces are different, the detection rates are different for each trace and it is impossible to set a fixed threshold for all the websites.

Even if Wehe seems efficient, we want to point out that it actually does not play and compare two streamed flows as if they were going to be displayed, the nominal and the differentiated. It tests a pre-recorded trace that is sent without differentiation, and also sent potentially in a “deteriorated”

way, but not interacting with the ABR algorithm. A challenge in a near future would be to imagine a procedure comparing two streaming videos *played* with ABR: an encrypted one (normally not differentiated), and a nominal one (potentially differentiated).

Another issue not discussed in this paper but worth further investigation is the detection test done in Wehe. What is implemented is a modification of the rigorous Kolmogorov-Smirnov test which looks at the difference between two empirical cumulative distribution functions of observed “instantaneous” throughputs and tests if it is statistically significant. Wehe modifies this test by using a variant with less theoretical foundations, based on heuristics, but probably conservative to avoid problematic false positives. Designing theoretically-proven and robust tests is one of the goals of our future research.

REFERENCES

- [1] T. Lenard and R. E. May, *Net Neutrality or Net Neutering: Should Broadband Internet Services be Regulated*. Springer, 2006.
- [2] L. Belli and P. E. De Filippi, *Net neutrality compendium: Human rights, free competition and the future of the Internet*. Springer, 2016.
- [3] P. Maillé and B. Tuffin, *From Net Neutrality to ICT neutrality*. Springer, 2022.
- [4] X. Castoreo, P. Maillé, and B. Tuffin, “Weaknesses and Challenges of Network Neutrality Measurement Tools,” in *Proc. of 16th IEEE International Conference on Network and Service Management (CNSM)*, Virtual Conference, Apr. 2020. [Online]. Available: <https://hal.inria.fr/hal-02542689>
- [5] T. Garrett, L. E. Setenareski, L. M. Peres, L. C. E. Bona, and E. P. Duarte, “Monitoring network neutrality: A survey on traffic differentiation detection,” *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 2486–2517, 2018.
- [6] A. Molavi Kakhki, A. Razaghpanah, A. Li, H. Koo, R. Golani, D. Choffnes, P. Gill, and A. Mislove, “Identifying Traffic Differentiation in Mobile Networks,” *Proceedings of the 2015 Internet Measurement Conference*, pp. 239–251, October 2015.
- [7] Northeastern University Wehe Project, “Wehe,” <https://play.google.com/store/apps/details?id=mobi.meddle.wehe> on Google Play Store, accessed March 5, 2020.
- [8] X. Castoreo, P. Maillé, and B. Tuffin, “Analyzing the Wehe Network Neutrality Monitoring Tool,” in *Proc. of GECON*, 2021. [Online]. Available: <https://hal.inria.fr/hal-03177366>
- [9] The Linux Foundation, “netem,” <https://wiki.linuxfoundation.org/networking/netem>, last accessed 23 Oct 2020.