Studying Forwarding Differences in European Mobile Broadband with a Net Neutrality Perspective

Enrico Gregori, Valerio Luconi
IIT-CNR, Pisa, Italy
Email: enrico.gregori@iit.cnr.it, valerio.luconi@iit.cnr.it

Alessio Vecchio
Dip. di Ingegneria dell'Informazione, University of Pisa, Italy
Email: alessio.vecchio@unipi.it

***Abstract*—Availability of deep packet inspection methods and systems allows network operators to classify traffic on the base of the application type. Once classified, traffic may be subject to artificial bandwidth limitations (e.g. in case of resource-demanding applications) or to class-dependent forwarding policies (e.g. to divert the traffic generated by specific applications on low-priority links). In this paper we describe a method that can be useful to detect the presence of class-dependent forwarding policies. The method is based on traceroute-like mechanisms embedded within the normal communication flow of an application. The method is contextualized in a study about the neutrality of mobile network operators, to understand if a correlation can be found between the presence of class-dependent forwarding strategies and limitations of bandwidth.**

***Index Terms*—Traceroute, net neutrality, network measurement.**

I. Introduction

The network neutrality principle states that an operator should provide the same treatment to all traffic flowing through its networks. Studies have demonstrated that a neutral Internet is possibly a benefit for the expansion of ISPs [1], [2]. In the past few years several neutrality violations have been reported, such as blocking or degrading the performance of specific applications and services (e.g., peer-to-peer, video streaming, VoIP). This led to the adoption of specific rules to protect the neutrality of the Internet [3], [4]. In the EU, net neutrality regulations are also considered one of the major achievements towards an open Internet. ISPs are not allowed to block or throttle traffic. Differentiation can be enforced only in exceptional circumstances, such as preserving the integrity of the network, or when a congestion occurs.

Several methods have been proposed to assess the neutrality of ISPs. The vast majority of them focused on the performance experienced by different classes of traffic in terms of measured throughput. The most studied application has been BitTorrent, as it is particularly resource-demanding from the point of view of providers [5], [6]. While there are numerous studies that focus on the wired part of the Internet, only a few are aimed at assessing neutrality in a mobile broadband scenario.

In addition, modern firewalls and traffic shapers include sophisticated capabilities, such as the so called policy routing, which gives the possibility to assign different forwarding paths to different types of traffic. This feature could be used to forward certain classes of traffic on slower or more congested paths, thus indirectly causing differentiation in terms of network performance.

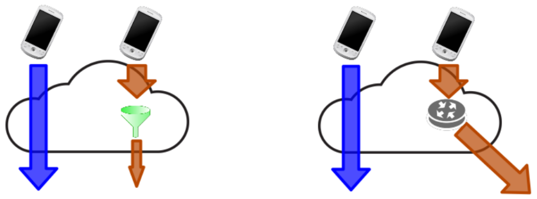


Fig. 1: Differentiation in terms of bandwidth and/or path.

In this paper we study the presence of differences, in terms of paths, for some classes of traffic. The main contribution of this paper is the development of a technique useful to collect the path traversed by different classes of traffic (at the application level). We extended NeutMon [7], a tool able to detect class-dependent bandwidth limitations, to include also this capability. The idea is to possibly discover intentional differentiation in terms of forwarding strategies. In addition we present a method useful to analyze the experimental results obtained with our tool. Preliminary experiments have been carried out on a set of European mobile network operators using the MONROE testbed [8].

The rest of the paper is structured as follows. Section II describes the design, implementation and validation of the presented tool. Section III describes the procedure used to analyze experimental data. In Section IV we show the experimental results. Finally, Section V describes the state of the art, and Section VI concludes the paper.

II. NeutMon

NeutMon is a tool aimed at detecting violations of net neutrality in the path between two endpoints (a client and a server). The first version of the tool was able to detect differentiation only in terms of throughput. NeutMon has been extended to support detection of differentiation in terms of forwarding rules, as explained in the rest of the paper. Two types of tests can be executed: Speed test and Traceroute test. Each test is aimed at detecting differentiation between two classes of traffic: BitTorrent traffic (BT) and Random traffic. Since random traffic is used as a baseline, we will refer to it as Control traffic (CT). The Speed test measures the application-level throughput obtained by BT and CT between the client

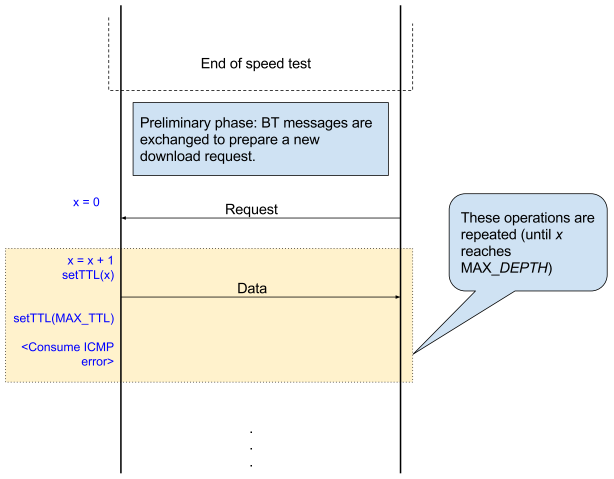

End of speed test
Preliminary phase: BT messages are exchanged to prepare a new download request.
x = 0
Request
x = x + 1
setTTL(x)
Data
setTTL(MAX_TTL)
<Consume ICMP error>
These operations are repeated (until x reaches MAX_DEPTH)

Fig. 2: Example of the procedure for discovering the path.

Algorithm 1 Traceroute test procedure.

```
 1: $MAX\_TTL \leftarrow$ System default TTL
 2: $MAX\_DEPTH \leftarrow 30$
 3:
 4: for all $x \in \{1..MAX\_DEPTH\}$ do
 5:     setTTL($x$)
 6:     start timer
 7:     send 100 bytes
 8:     setTTL($MAX\_TTL$)
 9:     while $True$ do
10:         try
11:             listen for and consume ICMP Time Exc. errors
12:             if ICMP Time Exceeded errors arrive then
13:                 restart timer
14:             end if
15:         catch timer expired
16:             break
17:         end try
18:     end while
19: end for
```

and the server. The Traceroute test instead collects the path traversed by BT and CT. The reference scenario is exemplified in Figure 1. Both tests are implemented on top of TCP and are executed in two directions: uplink and downlink (with respect to the client). NeutMon is implemented in Python and released as open source code[1].

A. Speed test

The test runs a data transfer between client and server with the two classes of traffic: BT and CT. The former follows the BitTorrent Protocol Specification [9], to be sure that it is recognized as a real BitTorrent transfer by possible shapers along the path. BT traffic starts with a handshake phase that identifies the two speakers as BitTorrent speakers and then the actual data transfer is executed (and the throughput measured). CT follows the same pattern, but with completely random content. Uplink and downlink phases are symmetrical, with reversed roles of client and server. The throughput is measured by the endpoint that is receiving the data, and the results are always stored on the server. The reader is forwarded to [7] for a more detailed description.

B. Traceroute test

The Traceroute test implemented in NeutMon operates according to the principles of a typical traceroute application, however it also introduces some important novelties. Classic traceroute sends IP probes (with either UDP, ICMP, or TCP as transport protocol) without establishing a connection with the target host. If the transport protocol is UDP, probes have empty or random payload. If ICMP, the probes are echo requests. If TCP, the probes are TCP SYNs. Traceroute cyclically sends probes with increasing IP TTL values (starting from 1), until the target host is reached, or until a certain TTL value is reached (hereafter MAX_DEPTH). For common traceroute implementations $MAX\_DEPTH = 30$. At each iteration, the probe can reach either an intermediate router or the target host. When an intermediate router is reached, it should send back an ICMP Time Exceeded packet. If the target is reached, it should respond with an ICMP Port Unreachable, ICMP Echo Reply, or a TCP RST/SYN+ACK, depending on the type of probes (UDP, ICMP, and TCP respectively). These messages stop the traceroute operations. An enhancement to standard traceroute is provided by Paris traceroute, which handles the presence of anomalies and load balancing by keeping fixed the fields that are used by routers to balance traffic (i.e. the 5-tuple) [10].

In NeutMon the Traceroute test is executed after the Speed test using the same connection. This is a substantial difference with respect to the standard traceroute, which does not open a connection, as we want the Traceroute test traffic to be categorized as application level traffic (BT or CT). In the following we describe the uplink version of the traceroute test, as downlink is symmetrical. The traceroute test starts with a preliminary phase that consists of some BitTorrent messages aimed at commanding a new download operation. In particular the client sends an *unchoke* message to tell the server that it is allowed to send data requests. The server sends back an *interested* message and the request for data chunks. The client then sends back the requested data. The data transfer includes the traceroute-like mechanisms. In particular, before starting, the endpoint sets to 0 the value of a variable x. Such variable is used to detect the router at x hops from the client (left-hand side in Figure 2). Data is sent according to the following procedure: i) variable x is incremented; ii) the TTL associated to the socket is set to x; iii) data is sent through the socket, this data will elicit an ICMP error on the router at x hops from the sender; iv) the TTL associated to the socket is reset to its default value (MAX_TTL), this is done to re-transmit the same data with a TTL that makes it reach the other endpoint; v) ICMP errors are consumed.

This procedure is repeated until x reaches the maximum exploration depth (MAX_DEPTH). The traceroute test is then executed again in the opposite direction (the two endpoints

[1]http://vecchio.iet.unipi.it/neutmon/

play swapped roles). As for the Speed test, the Traceroute test for CT follows the same pattern as BT but with random payloads. The entire procedure is specified in Algorithm 1. ICMP Time Exceeded errors are consumed within a cycle to handle possible TCP re-transmissions. Execution exits from the `while` cycle when a timer associated to the socket expires (`catch`). In other words, if no ICMP Time Exceeded errors are received for a given amount of time, the tool assumes that the router at the considered hop count is not responding and it proceeds to the next hop.

It must be noticed that since NeutMon establishes a connection with the target and at each step resets TTL to MAX_TTL, the tool is not able to detect when the target is reached. Thus, the algorithm continues until MAX_DEPTH is reached.

C. Implementation

NeutMon has been implemented to run on the MONROE platform [8]. MONROE is a large-scale and distributed testbed aimed at supporting experimentation on mobile broadband networks. MONROE counts over 400 nodes distributed in four European countries: Italy, Norway, Spain, and Sweden. MONROE nodes host SIM cards of 12 mobile network operators belonging to these countries. The SIM cards of one Italian operator, Vodafone, are also hosted by Spanish nodes, in which they operate in roaming. Each node is divided into two units (head and tail) that together can host up to three SIM cards of separate operators. Nodes are either static or mobile. The latter ones can be installed on trains, trucks, or buses. For security reasons MONROE users do not have direct access to node resources. Execution of users' code is based on containerization, which is a form of OS-level virtualization. A container can access only the resources and devices that the kernel is assigning to it. The chosen containerization software is docker [11], thus users have to pack all their experiment software in a docker container in order to execute measurements on MONROE nodes. The container is then scheduled for execution on the nodes chosen by the user via either a web interface or a command line client. The operating system available on MONROE nodes is Debian Linux. MONROE nodes provide a number of Python-based libraries and runtime services. To reduce the footprint of the container image hosting NeutMon, the latter is written in Python (images are transferred onto nodes before executing experiments, and a small image can be deployed in shorter time and with reduced traffic).

D. Validation

We validated the Traceroute test functionalities in a controlled environment hosted between the IIT-CNR and the Dipartimento di Ingegneria dell'Informazione of the University of Pisa.

The client machine was located in the IIT-CNR network, which hosts a Palo Alto firewall [12]. Such device is capable of recognizing traffic at the application level via deep packet inspection, and blocking, shaping and forwarding traffic

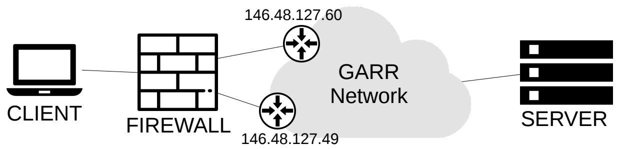


Fig. 3: Validation architecture.

according to policies defined by administrators. The server machine was located at the University of Pisa. The networks of the IIT-CNR and of the University of Pisa are connected via the GARR network (the Italian research network) [13], a high speed network whose link capacity is 20 Gbps and more. The IIT-CNR and the University of Pisa networks run instead at 1 Gbps.

The validation scenario is shown in Figure 3. The Palo Alto firewall can sort traffic between two router hops. We set up the Palo Alto firewall to assign to BT a different path from the other types of traffic. In particular BT was sent to interface 146.48.127.60 and all other traffic was sent to interface 146.48.127.49. We executed ten runs to ensure reliability, and in all of them NeutMon was able to identify the correct path for both BT and CT. We also executed ten runs with BT traffic shaped to 4 Mbps (in addition to the different forwarding policy). NeutMon was able to detect the correct path for both traffic types and detect the shaping of BT (the measured throughput for BT was always 4 Mbps, while for CT it was on average 689 Mbps).

III. Comparing traceroute results

The presence of differences in traceroute results is not always due to differentiation. For instance, two different paths, starting from the same source and ending in the same target, may be characterized by different intermediate nodes as a consequence of load balancing [10]. Thus, to detect the presence of path differentiation based on the class of traffic we decided to not analyze a single traceroute for the two classes. Instead, we collect a set of traceroute results for each class and then we compare the two sets to highlight statistically relevant differences (i.e. that could not be due to random load balancing).

Let $P^C(s,d) = [i_1, i_2, ..., i_n]$ be the path between source s and destination d detected when using traffic belonging to class C, where i_j is the interface found at the jth hop (also indicated as $P^C(s,d)[j]$). We define $\mathcal{P}^C(s,d) = \{P_1^C(s,d), P_2^C(s,d), ..., P_t^C(s,d)\}$ the set of t traceroute measurements between source s and destination d collected using traffic belonging to class C. In the following, for the sake of clarity, we omit source and destination, thus we will refer to the sets collected when using BT and CT simply as $\mathcal{P}^{BT}$ and $\mathcal{P}^{CT}$. For each hop $j \in \{1..n\}$, we compute the set of interfaces found when using BT and CT: $\mathcal{P}^{BT}[j] = \{P_1^{BT}[j], P_2^{BT}[j], ..., P_t^{BT}[j]\}$ and $\mathcal{P}^{CT}[j] = \{P_1^{CT}[j], P_2^{CT}[j], ..., P_t^{CT}[j]\}$. Then we compute the difference between the two sets at each hop. In particular, we

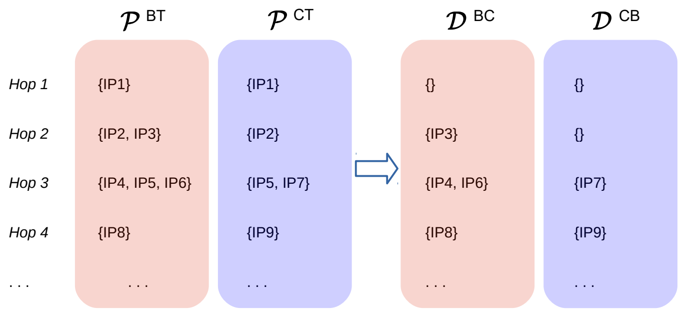


Fig. 4: Example of the procedure used for comparing traceroute results.

TABLE I: Countries and operators included in experiments.

Country	Operator
Italy	TIM
	Vodafone
	Wind
Norway	ICE Nordisk
	Telenor
	Telia Mobile
	Telia Norge
Spain	Orange
	Vodafone (roaming)
	Yoigo
Sweden	H3G Access
	Telenor
	Telia Mobile

compute $\mathcal{D}^{BC}[j] = \mathcal{P}^{BT}[j] \setminus \mathcal{P}^{CT}[j]$ and $\mathcal{D}^{CB}[j] = \mathcal{P}^{CT}[j] \setminus \mathcal{P}^{BT}[j]$. The former is the set of interfaces found at hop j by BT and not found by CT, the latter is the set of interfaces found at the same hop by CT and not found by BT. We used the same procedure to compare the results obtained by BT and CT against Paris traceroute. Thus, we also computed $\mathcal{D}^{BP}[j] = \mathcal{P}^{BT}[j] \setminus \mathcal{P}^{Paris}[j]$ and $\mathcal{D}^{CP}[j] = \mathcal{P}^{CT}[j] \setminus \mathcal{P}^{Paris}[j]$.

If the cardinality of $\mathcal{D}^{BC}$ and/or $\mathcal{D}^{CB}$ at a given hop is large, we may reasonably state that such difference could not be due to just random load balancing. In other words, the class of traffic may play a role in the forwarding decisions taken at previous hops. In the same way, the presence of large values for the cardinality of $\mathcal{D}^{BP}$ and/or $\mathcal{D}^{CP}$ could reveal the presence of forwarding policies based on the transport protocol.

For example, let us suppose that these three traceroute results have been obtained for BT:
$[IP1, IP2, IP4, IP8]$,
$[IP1, IP2, IP5, IP8]$,
$[IP1, IP3, IP6, IP8]$
and these for CT:
$[IP1, IP2, IP5, IP9]$,
$[IP1, IP2, IP7, IP9]$,
$[IP1, IP2, IP5, IP9]$.
The resulting sets would be the ones shown in Figure 4.

IV. Results

We ran a preliminary measurement campaign in November-December 2017. We tested the operators reported in Table I. For each operator twelve measurements have been carried out. In each measurement, we collected the path using the two classes of traffic (BT and CT). We also collected the path towards the destination using Paris traceroute with the MDA algorithm, which should enumerate all possible paths between a source and a destination [14]. The target, for all experiments, was a server hosted by the University of Pisa, Italy.

Figure 5 shows the plots concerning the Traceroute test uplink direction (from the MONROE node to the server). Each plot depicts the cardinality of $\mathcal{D}^{BC}[j]$ and $\mathcal{D}^{CB}[j]$ against the hop number j. The upper half of the plot shows the values of $\mathcal{D}^{BC}[j]$ cardinality (y-axis) at each hop j (x-axis). The lower half instead shows the values for $\mathcal{D}^{CB}[j]$. The farther the points from the middle line of the plot, the bigger the difference in terms of number of interfaces between the two flows. Large values could represent hops where different forwarding rules are applied on the base of the class of traffic. The plots also show for each hop j the cardinality of $\mathcal{D}^{BP}[j]$ and $\mathcal{D}^{CP}[j]$.

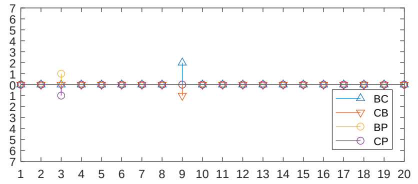
For some operators we did not observe any difference, in terms of path, when using BT or CT. We did not observe any difference also depending on the tool used for collecting the path (NeutMon and Paris traceroute). This happened with Vodafone (both non roaming and roaming), Orange, and ICE Nordisk. In addition Yoigo uplink traceroutes always failed for both BT and CT. Since all these operators show no difference, their plots are not included.

In Italy, TIM and Wind show forwarding differences between BT and CT. TIM seems to show some minor differences at hop 9, which still belongs to the TIM access network (we mapped the IPs to the owner's network using public Internet Routing Registries data [15]). These differences are minimal and could be due to load balancing. Wind shows a greater difference at hops 4 and 7, which are still in the Wind network. The amplitude of difference could be compatible with forwarding policies based on the class of traffic.

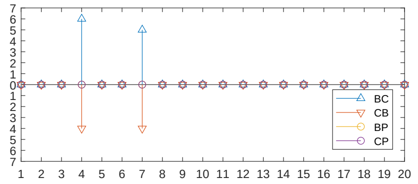
In Norway, traceroutes show differences for three operators: Telenor, Telia Mobile, and Telia Norge. The most evident differences between BT and CT however are not inside the access operator's network. Instead, they occur within an upstream provider that all the three have in common. Such upstream provider is Telianet, autonomous system number 1299. The network of this upstream provider starts at hop 10 for Telenor, hop 6 for Telia Mobile, and hop 7 for Telia Norge.

In Sweden, all the three operators show traceroute differences. For H3G the differences are found in the operator's network, and this could indicate that a different path is assigned to the different classes of traffic. The other two operators show a similar behavior to Norwegian ones. The differences are present but in an upstream provider, which again is Telianet, whose network starts at hop 13 for Telenor, and at hop 5 for Telia Mobile.

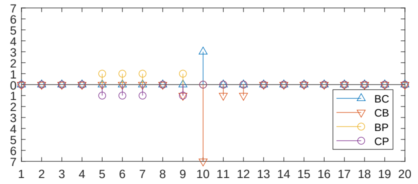
We can also observe that in some cases there are some differences between the paths experienced by BT and CT and the paths experienced by Paris traceroute. This happens mainly in



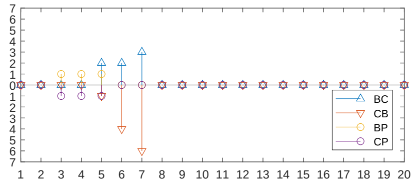
(a) Italy, TIM



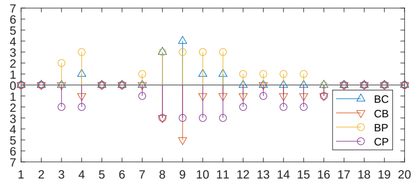
(b) Italy, Wind



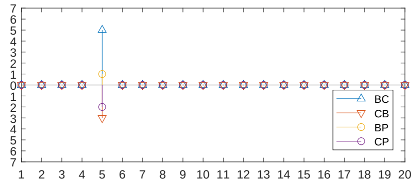
(c) Norway, Telenor



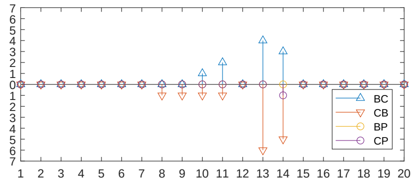
(d) Norway, Telia Mobile



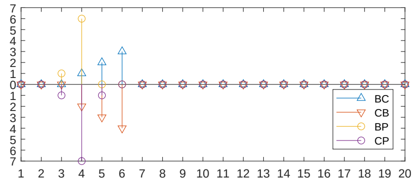
(e) Norway, Telia Norge



(f) Sweden, H3G Access



(g) Sweden, Telenor



(h) Sweden, Telia Mobile

Fig. 5: Uplink traceroute comparison. The plots show the cardinality of $\mathcal{D}$ (y-axis) against the hop count j (x-axis).

Norwegian and Swedish operators. In particular the differences are quite large in Telia Norge and Telia Mobile Sweden. Again, the upstream provider in these cases is Telianet. This could indicate that the provider forwards monitoring traffic (such as traceroute) on different paths, or that it selects different paths according to the transport protocol (Paris traceroute uses UDP probes).

Our results show that some operators seem to apply forwarding rules that may take the class of traffic into account. For others no differences could be detected from this point of view. We now compare these results with the results presented in [7], which are relative to the performance in terms of bandwidth obtained by BT and CT with the same operators. In [7] we showed that one operator (Vodafone) applied differentiation in terms of throughput experienced by BT and CT (both in the home network and in roaming). In particular CT obtains a much better performance. Two other operators (Yoigo and Telenor Sweden) obtained poor performance for both BT and CT. All other operators did not show any differences in terms of performance. As stated above, for Vodafone no differences in terms of forwarding are observed. For Yoigo, all uplink traceroutes (for both BT and CT) failed. The low performance in terms of bandwidth could be related to the cause of the traceroute failures (but further investigation is needed). For Telenor Sweden we observed forwarding differences in an upstream provider (Telianet). We believe that this is not the cause of the poor performance, as the same upstream provider shows forwarding differences also with other operators, but in these cases no performance degradation is observed.

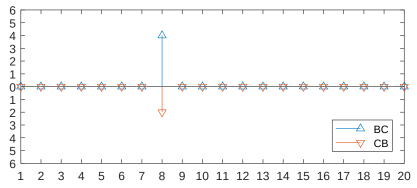
All other operators that showed forwarding differences did not show a corresponding differentiation in performance (as reported by [7]). It must be noticed that the vast majority of mobile network operators assign to their users private IP addresses (i.e., addresses belonging to the networks 10.0.0.0/8, 192.168.0.0/16, and 172.16.0.0/12 [16]), and provide access to the Internet via Network Address Translation (NAT). In this procedure the IP addresses fields of IP header and the port fields of TCP header can be changed, thus load balancing could be triggered on the traversed path, since these fields are frequently used for load balancing [10]. Thus, we strongly believe that further investigation is needed to understand if the presence of NATs is related to the results here shown.

Figure 6 shows the same plots for the downlink direction, i.e. from the server to the MONROE node. The figure includes only the plots of operators that showed some differences (even if minimal). The only operators that showed substantial differences are Telia Mobile Norway and Telia Mobile Sweden. Again the differences are found in the Telianet network, which is an upstream provider of both these mobile operators.

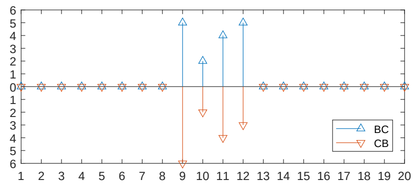
V. Related work

The vast majority of the techniques and tools for assessing net neutrality focus on measuring the performance obtained by different classes of traffic in terms of throughput, loss and latency [17].

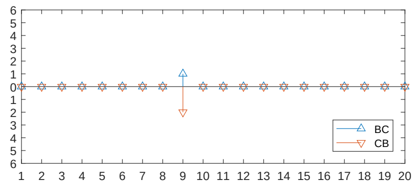
Among these the most relevant are Glasnost and Neubot, which share the approach of relying on the contribution of volunteer users. Glasnost is a system aimed at detecting differentiation in ISPs [18]. The client is implemented as a Java applet, thus taking measurements does not require complex installations, just a web browser. The test compares the throughput of several applications (including BitTorrent) with



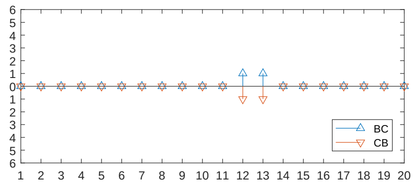
(a) Italy, Wind



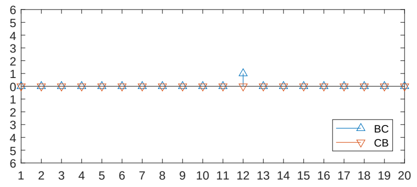
(b) Norway, Telia Mobile



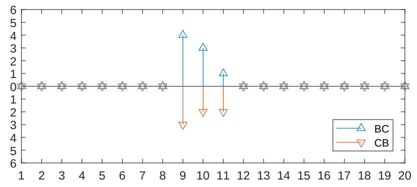
(c) Norway, Telia Norge



(d) Spain, Orange



(e) Sweden, Telenor



(f) Sweden, Telia Mobile

Fig. 6: Downlink traceroute comparison. The plots show the cardinality of $\mathcal{D}$ (y-axis) against the hop count j (x-axis).

random flows with similar characteristics. New applications can be easily added by recording their traces and submitting them to the system. Neubot aims at measuring the neutrality of the Internet [19]. Neubot operates in the background and periodically runs tests to measure the quality of service of different applications. Both systems rely on servers hosted on the M-Lab infrastructure [20]. Other works based on active measurements are NetPolice [21], ShaperProbe [22], and DiffProbe [23].

Network neutrality in a mobile broadband environment has been recently considered by few works. First, the tool BonaFide has been developed with the aim of detecting differentiation on several application protocols, such as BitTorrent, HTTP, Flash video, Real Time Streaming Protocol, VoIP H323, Session Initiation Protocol (SIP) [24]. Another method specifically designed to operate in a wireless environment is discussed in [25]. The work used an approach based on VPN. The VPN is used to record a trace of networked applications run on the user's smartphone. The trace is replayed and compared with the same trace encrypted in a tunnel (which should not be affected by differentiation). The method has been implemented in a publicly available Android app. Results showed that several US mobile network operators apply traffic differentiation policies.

The ability of traceroute in discovering Internet paths has been used by researchers to infer the interconnections between the entities that compose the Internet. Several systems based on traceroute measurements have been deployed, such as CAIDA Ark [26], [27], iPlane [28], DIMES [29], or Portolan [30]. However, as far as we know, traceroute has never been used to for detecting traffic differentiation.

Several variations of the traceroute mechanisms have been implemented, for improving its effectiveness or for adding new functionalities. The most relevant example is Paris traceroute [10] which is designed to solve known issues caused by the presence of load-balancers. An extension of Paris traceroute, named MDA (Multipath Detection Algorithm), has been implemented to discover all the paths generated by load-balancing mechanisms [14]. Another relevant extension to traceroute is tracebox, which adds to traceroute support for discovering middleboxes (i.e., machines that operate at levels higher than the network level along the path between a source and a destination) [31].

To the best of our knowledge this is the first work where a traceroute-like mechanism is used to detect, at the application-level, differences in terms of forwarding strategies.

VI. Conclusion

In this paper we presented a traceroute-like mechanism useful for detecting the presence of differences in the paths followed by different classes of traffic. We embedded the mechanism in NeutMon, a system aimed at studying neutrality of operators in terms of both performance and forwarding policies.

We used the MONROE testbed to study the path assigned to BitTorrent traffic and random traffic in a set of European mobile network operators spread in four countries. Preliminary results show that some operators seem to forward the two types of traffic on different paths. However this practice is not reflected in a degradation of performance (when comparing the results presented here with the ones in [7]).

For other operators, the paths show significant differences only when entering in upstream providers' networks. This seems to indicate that even if the possibility of differentiating the forwarding path according to the application-level class of traffic exists, it is not used to degrade the performance of applications, but rather to perform simple traffic engineering.

Future work will concern an extensive campaign of measurements, to better understand the phenomena emerging from these preliminary experiments.

Acknowledgment

This work is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 644399 (MONROE) through the second open call for projects. The views expressed are solely those of the authors.

References

[1] J. Pil Choi and B.-C. Kim, "Net neutrality and investment incentives," *The RAND Journal of Economics*, vol. 41, no. 3, pp. 446–471, 2010.

[2] A. Antonopoulos, E. Kartsakli, C. Perillo, and C. Verikoukis, "Shedding Light on the Internet: Stakeholders and Network Neutrality," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 216–223, 2017.

[3] B. Obama, "Net Neutrality: President Obama's Plan for a Free and Open Internet," https://obamawhitehouse.archives.gov/node/323681, 2008, [Online; accessed 24-February-2018].

[4] "BEREC Guidelines on the Implementation by National Regulators of European Net Neutrality Rules," http://berec.europa.eu/eng/document_register/subject_matter/berec/download/0/6160-berec-guidelines-on-the-implementation-b_0.pdf, 2016, [Online; accessed 6-october-2017].

[5] N. Weaver, R. Sommer, and V. Paxson, "Detecting Forged TCP Reset Packets," in *Proceedings of the Network and Distributed System Security Symposium, NDSS, San Diego, California, USA*, 2009.

[6] M. Dischinger, A. Mislove, A. Haeberlen, and K. P. Gummadi, "Detecting Bittorrent Blocking," in *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '08. New York, NY, USA: ACM, 2008, pp. 3–8.

[7] E. Gregori, V. Luconi, and A. Vecchio, "NeutMon: Studying Neutrality in European Mobile Networks," in *Proc. CNERT '18, to appear*, 2018.

[8] Ö. Alay, A. Lutu, R. García, M. Peón-Quirós, V. Mancuso, T. Hirsch, T. Dely, J. Werme, K. Evensen, A. Hansen, S. Alfredsson, J. Karlsson, A. Brunstrom, A. S. Khatouni, M. Mellia, M. A. Marsan, R. Monno, and H. Lonsethagen, "Measuring and assessing mobile broadband networks with MONROE," in *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2016, pp. 1–3.

[9] B. Cohen, "The BitTorrent Protocol Specification," http://www.bittorrent.org/beps/bep_0003.html, 2017, [Online; accessed 6-october-2017].

[10] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira, "Avoiding Traceroute Anomalies with Paris Traceroute," in *Proc. ACM SIGCOMM IMC '06*, 2006, pp. 153–158.

[11] "Docker," https://www.docker.com/.

[12] "Palo Alto Networks," https://www.paloaltonetworks.com/.

[13] "Consortium GARR," https://www.garr.it/.

[14] B. Augustin, T. Friedman, and R. Teixeira, "Multipath tracing with Paris traceroute," in *Proc. IEEE/IFIP E2EMON '07*, 2007, pp. 1–8.

[15] "IRR - Internet Routing Registry," http://www.irr.net/.

[16] R. G. Moskowitz, D. Karrenberg, Y. Rekhter, E. Lear, and G. J. de Groot, "Address Allocation for Private Internets," RFC 1918, 1996.

[17] 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. PP, no. 99, pp. 1–1, 2018.

[23] ——, "Diffprobe: Detecting ISP Service Discrimination," in *Proceedings of the 29th Conference on Information Communications*, ser. INFOCOM'10. Piscataway, NJ, USA: IEEE Press, 2010, pp. 1649–1657.

[18] M. Dischinger, M. Marcon, S. Guha, K. P. Gummadi, R. Mahajan, and S. Saroiu, "Glasnost: Enabling End Users to Detect Traffic Differentiation," in *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 27–27.

[19] S. Basso, A. Servetti, and J. C. D. Martin, "The network neutrality bot architecture: A preliminary approach for self-monitoring of Internet access QoS," in *2011 IEEE Symposium on Computers and Communications (ISCC)*, June 2011, pp. 1131–1136.

[20] C. Dovrolis, K. Gummadi, A. Kuzmanovic, and S. D. Meinrath, "Measurement Lab: Overview and an Invitation to the Research Community," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 3, pp. 53–56, Jun. 2010.

[21] Y. Zhang, Z. M. Mao, and M. Zhang, "Detecting Traffic Differentiation in Backbone ISPs with NetPolice," in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '09. New York, NY, USA: ACM, 2009, pp. 103–115.

[22] P. Kanuparthy and C. Dovrolis, "ShaperProbe: End-to-end Detection of ISP Traffic Shaping Using Active Methods," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, ser. IMC '11. New York, NY, USA: ACM, 2011, pp. 473–482.

[24] V. Bashko, N. Melnikov, A. Sehgal, and J. Schönwälder, "BonaFide: A traffic shaping detection tool for mobile networks," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, May 2013, pp. 328–335.

[25] A. Molavi Kakhki, A. Razaghpanah, A. Li, H. Koo, R. Golani, D. Choffnes, P. Gill, and A. Mislove, "Identifying Traffic Differentiation in Mobile Networks," in *Proceedings of the 2015 Internet Measurement Conference*, ser. IMC '15. New York, NY, USA: ACM, 2015, pp. 239–251.

[26] k. claffy, Y. Hyun, K. Keys, M. Fomenkov, and D. Krioukov, "Internet Mapping: From Art to Science," in *Proc. CATCH '09*, 2009, pp. 205–211.

[27] "The Cooperative Association for Internet Data Analysis Archipelago Measurement Infrastructure (CAIDA Ark)," http://www.caida.org/projects/ark/.

[28] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane: An Information Plane for Distributed Services," in *Proc. USENIX OSDI '06*, 2006, pp. 367–380.

[29] Y. Shavitt and E. Shir, "DIMES: Let the Internet Measure Itself," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, pp. 71–74, 2005.

[30] A. Faggiani, E. Gregori, L. Lenzini, V. Luconi, and A. Vecchio, "Smartphone-based crowdsourcing for network monitoring: Opportunities, challenges, and a case study," *IEEE Communications Magazine*, vol. 52, no. 1, pp. 106–113, January 2014.

[31] G. Detal, B. Hesmans, O. Bonaventure, Y. Vanaubel, and B. Donnet, "Revealing Middlebox Interference with Tracebox," in *Proc. IMC '13*, 2013, pp. 1–8.