Informing Protocol Design through Crowdsourcing Measurements

ANNA MARIA MANDALARI

EN CUMPLIMIENTO PARCIAL DE LOS REQUISITOS PARA EL GRADO DE DOCTOR EN

INGENIERÍA TELEMÁTICA

UNIVERSIDAD CARLOS III DE MADRID

DIRECTOR:

MARCELO BAGNULO

Mayo de 2019

PUBLISHED BY UNIVERSIDAD CARLOS III DE MADRID

First printing, March 2019

“The noblest pleasure is the joy of understanding.”

—Leonardo Da Vinci.

Acknowledgements

I WOULD LIKE TO BEGIN BY THANKING my advisor, Marcelo. Thanks for being so patient and gave me the possibility to develop my research skills. Thanks for hearing me out and supported me in any situations, included the most difficult ones (at work and not). I would never be where I am if you had not believed on my abilities. Thanks also to a great woman, an example for all women in computer science, Andra.

Thanks to David Clark, our conversation in 2015 was really inspiring and made me understand what I want to become in future.

Grazie alle mie amiche, sorelle da tutta una vita (letteralmente) Francesca e Valentina. Grazie per esserci sempre!

Un immenso grazie alla mia famiglia, senza il vostro supporto non so davvero come avrei fatto. Avete girato con me il mondo e ancora ne abbiamo di posti da vedere (madre non uccidermi!!). Grazie ai miei due fratelloni, alle mie cognate e a Melania, Dalila e Domi per essere così estremamente genuini e farmi ridere ogni volta che ne ho bisogno. Grazie a nonna Santella e Anna Maria per essere così profondamente "nonne" del Sud.

Gracias a todas las personas que he tenido el placer de conocer en estos 4 largos años en Madrizzz, desde peluquero hasta la chica del pan, pero un mas profundo gracias a: Miguel, eres extremamente sensible y bueno, un amigo como pocos; Juan, gracias para creer en mi y ayudarme siempre: Sofia, un ejemplo de mujer activa y insuperable; Olga, increíble las cosas que tenemos en común, gracias para no aburrirte nunca de escucharme; Alberto, solo tu, yo, los 35 grados de Madrid en Julio y dos raquetas; Angie, desde el instituto hasta Madrid , como si 10 años no hubieran pasado; Anibal, gracias para hacerme reír siempre; Domenico, il destino a volte ti fa incontrare persone come te, condividiamo la stessa città due volte; Rafael, gracias para creer en mis sueños y ayudarme a convertirlo en realidad.

Grazie ai miei amici di Reggio, sempre lì, sempre con me: Achille, Rosario e Giacomo.

Tack Andreas för att stödja mig när som helst.

Thanks to all those amazing people that I met during my internships in Barcelona and Norway: Ilias, Ozgu, Eleonora, Simone, Isa, Teodora, Matteo, Sasan, Johannes, Vangelis. Thanks for made me feel like at home, for the hiking, the drinks and the happiness.

Gracias a mis compañeros de la UC3M, por las cervezas, las noches en la Uni, las comidas y las risas. Gracias a las personas de la UC3M/IMDEA and other institutes con la cual he tenido el honor de trabajar: Paco, Alberto, Ruben, Narseo, Bob Briscoe y todos los staff técnico, especialmente Goyo, Rafa y Jose Antonio.

Gracias a Fernando para no perder las esperanzas conmigo y mi tenis.

Gracias a mi compañeros de METRICS, especialmente Roberto, Ines y Costas.

Gracias a mi fantastico team de Gustami italia en Madrid: David, Elisabetta, Alvaro, Francesco, Josef y Jorge.

Finally, thanks to the people from my new Lab in London: Hamed, Roman, Minos, Vincent, Yuting, Rania and Mohamed.

Grazie ad Anna, la mia amica "pazza" londinese and Andrew the most British man I've ever met.

Grazie mille al mio aragostino for believing in me.

PS. During my PhD I visited 17 different countries, took hundreds of flights, not slept for innumerable number of nights. I broke a finger and a leg, I run a half marathon, I changed 3 tennis rackets , went to 3 different horses centres, learnt how to snowboard, swam thousands of meters, fell in love hundreds of times and this is just the beginning (hopefully not for the bones ;))!

—Anna Maria Mandalari
London, February 2019

Published and Submitted Content

In compliance with the principles reffering to plagiarism in Law 14/2011 and in the code of good practices of the UC3M Doctoral School, in this Chapter a list the papers that are included in this thesis and have been published are included. The role of the author of the thesis for each of the contribution is also specified.

Chapter 4

- A. M. Mandalari, M. Bagnulo, and A. Lutu. Informing Protocol Design Through Crowdsourcing: The Case of Pervasive Encryption. *ACM SIGCOMM Computer Communication Review*, 45(4): 105–123, 2015b. doi: 10.1145/2787394.2787397. Mandalari et al. [2015b].
  - The contribution is fully included in the thesis.
  - The role of the author of this thesis is on developing the idea included in the paper for using crowdsourcing platform for TLS and HTTP/2 measurements. The author design the experimental setup, run the experiment and process the data for evaluation.
  - The material from this paper included in this thesis is not singled out with typographic means and references.

Chapter 5

- A. M. Mandalari, M. Bagnulo, and A. Lutu. TCP Fast Open: Initial Measurements. *ACM CoNEXT Student Workshop, Dec 2015, Heidelberg, Germany.*, 2015a. doi: 10.1145/2842665.2843561. Mandalari et al. [2015a].
  - The contribution is fully included in the thesis.
  - The role of the author of this thesis is on developing the idea included in the paper for using crowdsourcing platform for TCP Fast Open measurements. The author design the experimental setup, run the experiment and process the data for evaluation.
  - The material from this paper included in this thesis is not singled out with typographic means and references.

Chapter 6

- A. M. Mandalari, A. Lutu, B. Briscoe, M. Bagnulo, and O. Alay. Measuring ECN++: Good News for ++, Bad News for ECN over Mobile. *IEEE Communications Magazine*, 56(3):180–186, March 2018a. ISSN 0163-6804. DOI: 10.1109/MCOM.2018.1700739. Mandalari et al. [2018a].
  - The contribution is fully included in the thesis.
  - The role of the author of this thesis is on designing the experiments for evaluating ECN and ECN++ in fixed and mobile networks. The author run the experiments in different measurements platforms, process the data and evaluate the results.
  - The material from this paper included in this thesis is not singled out with typographic means and references.

Chapter 7

- A. M. Mandalari, M. A. D. Bautista, F. Valera, and M. Bagnulo. NATwatcher: Profiling NATs in the Wild. *IEEE Communications Magazine*, 55(3):178–185, March 2017a. ISSN 0163-6804. DOI: 10.1109/MCOM.2017.1600776CM. Mandalari et al. [2017a].
  - The contribution is fully included in the thesis.
  - The role of the author of this thesis is on building the tool for characterizing NATs in fixed lines. The author leverage a crowdsourcing platform for running the experiments, process the data and evaluate NATs characteristics.
  - The material from this paper included in this thesis is not singled out with typographic means and references.

Chapter 8

- A. M. Mandalari, A. Lutu, A. Dhamdhere, M. Bagnulo, and K. Claffy. Tracking the Big NAT across Europe and the U.S. abs/1207.0016, 2017b. URL `https://arxiv.org/abs/1704.01296`. Mandalari et al. [2017b].
  - The contribution is fully included in the thesis.
  - The role of the author of this thesis is on leveraging the Revelio tool for tracking CGNs in fixed lines. The author improve the tool, run the experiments in two crowdsourcing large scale measurement platforms and process the data.
  - The material from this paper included in this thesis is not singled out with typographic means and references.

Chapter 10

- A. M. Mandalari, A. Lutu, A. Custura, A. Safari Khatouni, O. Alay, M. Bagnulo, V. Bajpai, A. Brunstrom, J. Ott, M. Mellia, and G. Fairhurst. Experience: Implications of Roaming in Europe. In *Proceedings of the 24th Annual International Conference on*

Mobile Computing and Networking, MobiCom '18, pages 179–189, New York, NY, USA, 2018b. ACM. ISBN 978-1-4503-5903-0. DOI: 10.1145/3241539.3241577. Mandalari et al. [2018b].

- The contribution is fully included in the thesis.
- The role of the author of this thesis is on contributing on building a measurement platform for evaluating roaming in Europe. The author plays a key role on investigate the roaming configuration and its implication in terms of delay, DNS and HTTP performance.
- The material from this paper included in this thesis is not singled out with typographic means and references.

Abstract

Middleboxes, such as proxies, firewalls and NATs play an important role in the modern Internet ecosystem. On one hand, they perform advanced functions, e.g. traffic shaping, security or enhancing application performance. On the other hand, they turn the Internet into a hostile ecosystem for innovation, as they limit the deviation from deployed protocols. It is therefore essential, when designing a new protocol, to first understand its interaction with the elements of the path. The emerging area of crowdsourcing solutions can help to shed light on this issue. Such approach allows us to reach large and different sets of users and also different types of devices and networks to perform Internet measurements. In this thesis, we show how to make informed protocol design choices by expanding the traditional crowdsourcing focus from the human element and using crowdsourcing large scale measurement platforms.

We consider specific use cases, namely the case of pervasive encryption in the modern Internet, TCP Fast Open and ECN++. We consider such use cases to advance the global understanding on whether wide adoption of encryption is possible in today's Internet or the adoption of encryption is necessary to guarantee the proper functioning of HTTP/2. We target ECN and particularly ECN++, given its succession of deployment problems. We then measured ECN deployment over mobile as well as fixed networks. In the process, we discovered some bad news for the base ECN protocol—more than half the mobile carriers we tested wipe the ECN field at the first upstream hop. This thesis also reports the good news that, wherever ECN gets through, we found no deployment problems for the ECN++ enhancement. The thesis includes the results of other more in-depth tests to check whether servers that claim to support ECN, actually respond correctly to explicit congestion feedback, including some surprising congestion behaviour unrelated to ECN.

This thesis also explores the possible causes that ossify the modern Internet and make difficult the advancement of the innovation. Network Address Translators (NATs) are a commonplace in the Internet nowadays. It is fair to say that most of the residential and mobile users are connected to the Internet through one or more NATs. As any other technology, NAT presents upsides and downsides. Probably the most acknowledged downside of the NAT technology is that it introduces additional difficulties for some applications such as peer-to-peer applications, gaming and others to function properly. This is partially due to the nature of the NAT technology but also due to the diversity of behaviors of the different NAT implementations deployed in the Internet. Understanding the properties of the currently deployed NAT base provides useful input for application and protocol developers regarding what to expect when deploying new application in the Internet. We develop NATwatcher, a tool to test NAT boxes using a crowdsourcing-based measurement methodology.

We also perform large scale active measurement campaigns to detect CGNs in fixed broadband networks using NAT Revelio, a tool we have developed and validated. Revelio enables us to actively determine from within residential networks the type of upstream network address translation, namely NAT at the home gateway (customer-grade NAT) or NAT in the ISP (Carrier Grade NAT). We deploy Revelio in the FCC Measuring Broadband America testbed operated by SamKnows and also in the RIPE Atlas testbed.

A part of this thesis focuses on characterizing CGNs in Mobile Network Operators (MNOs). We develop a measuring tool, called CGNWatcher that executes a number of active tests to fully characterize CGN deployments in MNOs. The CGNWatcher tool systematically tests more than 30 behavioural requirements of NATs defined by the Internet Engineering Task Force (IETF) and also multiple CGN behavioural metrics. We deploy CGNWatcher in MONROE and performed large measurement campaigns to characterize the real CGN deployments of the MNOs serving the MONROE nodes.

We perform a large measurement campaign using the tools described above, recruiting over 6,000 users, from 65 different countries and over 280 ISPs. We validate our results with the ISPs at the IP level and, reported to the ground truth we collected. To the best of our knowledge, this represents the largest active measurement study of (confirmed) NAT or CGN deployments at the IP level in fixed and mobile networks to date.

As part of the thesis, we characterize roaming across Europe. The goal of the experiment was to try to understand if the MNO changes CGN while roaming, for this reason, we run a series of measurements that enable us to identify the roaming setup, infer the network configuration for the 16 MNOs that we measure and quantify the end-user performance for the roaming configurations which we detect. We build a unique roaming measurement platform deployed in six countries across Europe. Using this platform, we measure different aspects of international roaming in 3G and 4G networks, including mobile network configuration, performance characteristics, and content discrimination. We find that operators adopt common approaches to implementing roaming, resulting in additional latency penalties of ∼60 ms or more, depending on geographical distance. Considering content accessibility, roaming poses additional constraints that leads to only minimal deviations when accessing content in the original country. However, geographical restrictions in the visited country make the picture more complicated and less intuitive.

Results included in this thesis would provide useful input for application, protocol designers, ISPs and researchers that aim to make their applications and protocols to work across the modern Internet.

Contents

PART I FEASIBILITY OF NEW PROTOCOLS EXTENSIONS

PART II MEASURING NATS IN THE WILD

PART III IMPLICATION OF ROAMING IN EUROPE

List of Figures

1 Introduction

The indisputable success of the Internet and, consequently, the increasing demand from end-users for more secure and faster access to the online services drives the need for continuous innovation. Meeting these performance, security, and policy compliance requirements by designing new protocols or even optimizing existing ones is, however, challenging in today's Internet. Modern networks often rely on dedicated hardware components generically dubbed *middleboxes* to perform advanced processing functions like, for example, enhancing application performance (e.g., traffic accelerators, caches, proxies), traffic shaping (e.g., load balancers), optimizing the usage of IPv4 address space (e.g., NATs) or security (e.g., firewalls).

One major criticism of middleboxes is that they might filter traffic that does not conform to expected behaviors, thus ossifying the Internet and rendering it as a hostile environment for innovation. [1] It demonstrably becomes problematic to extend core Internet protocols, limiting the opportunities for optimization. For example, recent studies show that for IPv6 some intermediate nodes may inspect the contents of extension headers and discard packets based on the presence of unknown IPv6 options. [2] Moreover, it is widely acknowledged by the community that several of the protocols standardized by the IETF over the last few years including DCCP, UDP-lite, SCTP and several extensions to TCP, e.g. ECN and LEDBAT, face deployment challenges blamed on interference by middleboxes. [3]

This does not mean that it is impossible to deploy new protocols, but that in order to ensure success it is imperative to first understand the interaction of the proposed solutions with the middleboxes active along the path or how middelboxes behave. Recent studies [4], [5] on middleboxes behavior attempt to provide such information. However, the existing measurements use only a very small number of vantage points, e.g., only 142 measurement points are used. In order to perform representative Internet measurements and test realistic scenarios and different Autonomous Systems (ASes), what is missing is access to a high number of diverse vantage points. Also, more evi-

[1] M. Handley. Why the internet only just works. *BT Technology Journal*, 2006

[2] F. Gont et al. Transmission and processing of IPv6 options. Technical report, IETF draft-gont-6man-ipv6-opt-transmit-01, 2015

[3] https://www.ietf.org/mailman/listinfo/hops. *IRTF*, 2015

[4] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda. Is it still possible to extend TCP? In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 181–194. ACM, 2011

[5] B. Hesmans et al. Are TCP extensions middlebox-proof? In *Workshop on Hot topics in middleboxes and network function virtualization*. ACM, 2013

dences of the nature of the problems and the portions of the network in which they manifest are needed.

The emerging sea of crowdsourcing can be an efficient and quick solution to solve the problem. Until recently, large-scale Internet measurement infrastructures necessary to perform this type of analysis were available only to large Internet players, such as Google, Akamai and large ISPs. Consequently, the lack of public access to such resources makes it hard to repeat or verify their results. In an attempt to adjust the balance, several crowdsourcing large-scale measurement platforms became available to the research and operational community in the past years, e.g., RIPE Atlas [6], BISmark [7], SamKnows [8], PlanetLab [9], MONROE. [10]

Using crowdsourcing, group of people, institution and organizations can be engaged via a flexible open call for voluntarily completing a task. Crowdsourcing platforms can be large-scale measurement platforms such as the ones cited above, where some hardware is installed on people's home or institutions and no actions from the person owned the hardware is required or platform such as Microworkers [11] or Amazon's Mechanical Turk [12], where actions from people owned phones or personal computers are required.

By expanding the traditional crowdsourcing focus from the human element to use a diverse and numerous group of end-user devices as measurement vantage points [13], in this work, we leverage on crowdsourcing platforms to run Internet wide measurements. We demonstrated how to make informed protocol design choices by using a novel methodology for performing large scale Internet measurements, using crowdsourcing solutions approach.

While both crowdsourcing methodologies rely on Internet users performing a specific task (be that hosting custom equipment or performing custom tasks which require human intelligence), they are somewhat complementary. If in crowdsourcing large scale measurement platforms we have better control of the measurement agents, we are aware of the full specifications of the equipment being used and (in some cases) have more information about the environment where it is located, we are still limited by the number of vantage points and the types of tests that can be performed. In the case of crowdsourcing platforms (e.g, microworkers or amazon mechanical turks), we can leverage the access to a very high number of active users and their equipment. Though we lose in terms of control over the measurement agent, we gain in terms of number of available vantage points and the freedom of deploying our own custom-designed tests.

We exemplify next the efficiency of our methodology considering 7 case studies.

[6] V. Bajpai, S. J. Eravuchira, and J. Schönwälder. Lessons learned from using the ripe atlas platform for measurement research. *SIGCOMM Comput. Commun. Rev.*, 45(3):35–42, July 2015. ISSN 0146-4833

[7] F. Krueger and S. R. Andrews. Bismark: a flexible aligner and methylation caller for Bisulfite-Seq applications. *Bioinformatics*, 27(11):1571–1572, 04 2011. ISSN 1367-4803

[8] Samknows platform. https://www.samknows.com/. Accessed on 2019-02-03

[9] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman. Planetlab: An overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.*, 33 (3):3–12, July 2003b. ISSN 0146-4833

[10] O. Alay, A. Lutu, M. Peón-Quirós, V. Mancuso, T. Hirsch, K. Evensen, A. Hansen, S. Alfredsson, J. Karlsson, A. Brunstrom, A. Safari Khatouni, M. Mellia, and M. A. Marsan. Experience: An open platform for experimentation with commercial mobile broadband networks. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, MobiCom '17, pages 70–78, New York, NY, USA, 2017a. ACM. ISBN 978-1-4503-4916-1

[11] M. Hirth, T. Hoßfeld, and P. Tran-Gia. Anatomy of a crowdsourcing platform - using the example of microworkers.com. In *2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 322–329, June 2011. DOI: 10.1109/IMIS.2011.89

[12] M. Buhrmester, T. Kwang, and S. D. Gosling. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 6(1):3–5, 2011

[13] A. Doan et al. Crowdsourcing systems on the world-wide web. *ACM*, 2011

- The feasibility of pervasive encryption in the Internet ecosystem.
- TCP Fast Open (TFO).
- Measuring ECN++.
- Tracking and characterizing the Big NAT across Europe and the U.S.
- Profiling NATs in the wild.
- Characterizing Carrier Grade NATs in Mobile Broadband Networks.
- Measuring roaming in Europe.

Lessons learned from our work can help in designing robust protocol extensions. The first part of our work is devoted to test the extensions of new protocols in the Internet.

Through our methodology we tested the operation and functionalities against middleboxes of used protocols such as TLS (against different ports) and new protocols such as HTTP/2.

The public disclosure of the NSA global surveillance operations of U.S. citizens and foreign nationals generated a media frenzy, drawing a lot of attention towards the individual right for the confidentiality of communications in the digital era. The latest revelations on this topic acted as a catalyst for the urgency of increased privacy in the Internet. As a reaction from the operational Internet community, we now observe a stronger tendency to encrypt traffic over the Internet. [14] We have witnessed recently that many popular applications (e.g., web, Youtube video streaming) have migrated from HTTP to the HTTPS protocol. [15] The long-term objective of the research community is to provide encryption by default for all Internet communications.

However, before designing any solution, it is first essential to understand the feasibility of pervasive encryption in the Internet ecosystem by measuring the interaction of middleboxes with the TLS across the different TCP ports that currently use plain text protocols. In other words, we need to establish at this point whether using encryption in traditionally unsecured ports is even possible in today's Internet. With this goal in mind, we attempt to initiate TLS connections in 68 different ports that normally do not use any form of encryption and analyze the success of the connection. This is a first necessary step towards a full comprehension of the behavior of middleboxes relative to pervasive encryption.

On the other hand, in May 2015, version 2 of the Hypertext Transfer Protocol (HTTP/2) was standardized as RFC 7540. [16] HTTP/2

[14] S. Farrell et al. Pervasive monitoring is an attack. Technical report, RFC 7258, May, 2014

[15] D. Naylor et al. The cost of the "S" in HTTPS. In *ACM*, CoNEXT, 2014b

[16] M. Belshe, R. Peon, and M. Thomson. Hypertext Transfer Protocol Version 2 (HTTP/2). Technical report, RFC 7540, May, 2015

is an effort to reduce the gap between how modern websites are designed, and how their content is delivered. To do such, HTTP/2 introduces several novel features like header compression, full request and response multiplexing, and support for request prioritization and server push.

The IETF longly discussed about whether HTTP/2 should be encrypted by default using Transport Layer Security (TLS). Specifically, the standard defines that H2C should be negotiated by mean of an Upgrade header field in a classic HTTP/1 request.

Using crowdsourcing platform, over a period of one week we recruit more than 600 users distributed across 38 countries; 355 users perform measurements from fixed networks, while 322 users were only available on mobile. We build a complex dataset for a total of 140,432 connections. Results, from TLS and HTTP/2 campaigns, show that middleboxes affect H2C deployment, especially on port 80. H2C Upgrade requests are affected by proxies, failing to upgrade to HTTP/2.

Our methodology allows us to make an extensive study on how middleboxes behave in front of TCP new extensions, namely TCP Fast Open(TFO) and Expicit Congestion Notification++ (ECN++).

We measure in the wild whether TFO is supported by the Internet paths. To this end we recruit TFO users from Microworkers crowdsourcing platform. We create a tool called ExploreTFO that allows users to connect to our TFO server. Leveraging the access to these active users and their equipment we find that the 41,3% of the paths we test allowed for a successful TFO communication. We present a first step analysis of the deployability of TFO and we demonstrate that early results are not promising, reducing the possibility to decrease the latency due the TCP handshake.

Expicit Congestion Notification (ECN) is a way to mark packets to indicate that the capacity of a link is approaching exhaustion. ECN was standardized as a straight replacement for loss signals, but increasingly ECN is also being recognized as critical for low delay. [17]

A new proposal called ECN++ [18] proposes safe ways to remove all the original prohibitions on using ECN on each type of TCP packet. As with any new protocol, ECN++ can experience deployment problems, either because existing networks and servers protect themselves against out-of-the-ordinary behavior, or because optimizations have been built around a narrow and unchanging interpretation of the way protocols work. [19]

Therefore, this study sets out to measure how much existing networks and servers would mangle or block the ECN++ updates to TCP/IP. Using a crowdsourcing large scale measurement platform called MONROE, we test ECN and ECN++ support in both fixed and

[17] M. Welzl and G. Fairhurst. The Benefits of using Explicit Congestion Notification (ECN). Internet Draft draft-ietf-aqm-ecn-benefits-08, Internet Engineering Task Force, Nov. 2015. URL https://tools.ietf.org/html/draft-ietf-ecn-benefits. (Work in Progress)

[18] M. Bagnulo and B. Briscoe. ECN++: Adding Explicit Congestion Notification (ECN) to TCP Control Packets. Internet Draft draft-bagnulo-tcpm-generalized-ecn-04, Internet Engineering Task Force, May 2017. URL https://tools.ietf.org/html/draft-bagnulo-tcpm-generalized-ecn. (Work in Progress)

[19] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda. Is it still possible to extend TCP? In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 181–194. ACM, 2011

mobile networks. In particular, we test 18 mobile carriers, making it, to the best of our knowledge, the largest measurement study of ECN in mobile networks so far, even Apple had only tested three mobile carriers. [20] Although we only set out to measure ECN++ support, our measurements from mobile vantage points challenge the accepted belief that path traversal of ECN itself is free of problems.

The study does not just check for correct transitions of protocol fields; it also checks behavior. That is, it checks whether ECN-capable servers correctly reduce their rate in response to ECN feedback.

We discover that:

- More than half of the mobile carriers we tested bleach (clear) the ECN field at the first upstream IP hop. This contradicts the impression of hardly any ECN traversal problems that has been reinforced by all recent studies. Nonetheless, our testing with fixed connectivity is consistent with these previous studies.
- Bleaching ECN is benign[21], the connection continues, but without the benefits of ECN. We find no evidence of the ECN capability being blocked.
- Wherever ECN gets through, we find no problems for ECN++. We also find no problems with how servers respond to ECN-marking, but we do find some interesting congestion behaviors unrelated to ECN.

The second part of our work focuses on studying the characteristics and the behavior of such middleboxes. We centred our vision on Network Address Translation(NATs) with a deep analysis on Carrier Grade NATs (CGNs).

NATs were introduced back in the early 90s as a mean to cope with the incipient address depletion crisis. Together with Classless Interdomain Routing (CIDR), they successfully extended the lifetime of IPv4 from imminent depletion until very recently, when the Internet Assigned Numbers Authority (IANA) pool of IPv4 addresses finally ran out. [22] NATs are now a commonplace in the Internet and they are included by default in the Internet Access offerings for both residential and mobile customers.

NATs successfully extended the lifetime of the IPv4 indeed, but at a high cost: they hardcoded the client-server paradigm in the architecture of the Internet. The basic operation of a NAT relies on the creation of a mapping state between a private address and port pair and a public address and port pair. This state is created when a client using a private address initiates a communication with a server in the public Internet. Deploying applications that have an alternative paradigm, such as peer-to-peer applications, gaming or Voice-over-

[20] S. Cheshire. Networking for the Modern Internet. URL: https://developer.apple.com/videos/play/wwdc2016/714/, June 2016. (Presentation video: requires Safari browser)

[21] Nonetheless, if other links precede the cellular hop (e.g. a home router or bus/train connected over cellular), any CE-marking introduced in the home or vehicle network would be wiped, which would fool ECN sources into overrunning their local network.

[22] Potaroo Network Tool. http://www.potaroo.net/tools/ipv4/. Accessed on 2019-02-02

IP to name a few, that require hosts in the public Internet to initiate communications towards hosts behind a NAT is challenging and requires the use of the so-called NAT traversal techniques. These techniques are usually cumbersome and increase the latency, the traffic and the energy consumed by the endpoint.

While the aforementioned problem of supporting alternative application paradigms is fundamental to the nature of the NAT operation, it is exacerbated by the myriad of behaviors of the different NAT implementations deployed in the Internet. Different NATs use different criteria to create, preserve and remove their internal mapping state and have different filtering and forwarding rules. This severely complicates the job of applications willing to manage the NAT state in order to enable alternative communication models other than the client server one as they need to cope with all possible NAT flavours.

Another approach towards prolonging the life of current IPv4 address allocations is to deploy Carrier Grade NATs (CGNs), where Internet Service Providers (ISPs) share the same public IPv4 address across multiple end users. CGNs may introduce a number of issues for end users, service providers, content providers and government authorities. [23] There is some evidence that CGNs can cause dropped services in peer-to-peer applications, and lead to low performance of file transfer and video streaming sessions. [24] CGNs also introduce security challenges including traceability of IP addresses and anti-spoofing. Despite these challenges, CGNs offer an immediate relief to the IPv4 address scarcity problem, so it is likely that their popularity will increase over time. The use case for CGN differs in wireline vs. mobile networks. Given the rapid boost in the number of operational Internet-enabled mobile devices, and the scarcity of available IPv4 address space, mobile operators usually assign the same public IP address to multiple end users. Hence, some form of carrier-grade NAT technology has always been the norm, rather than the exception. [25] In wireline networks, however, end users are typically assigned public IPv4 addresses. While this situation may change as IPv4 addresses become increasingly scarce and ISP customer bases continue to grow, there is no source of systematic measurements of the prevalence or evolution of CGN deployments in ISPs.

While CGNs enable mobile operators to provide Internet-access to millions of devices with a limited amount of IPv4 addresses, their adoption imposes functional and potentially also performance penalties to the end-users that may affect their perceived quality of experience.

However, it is far from clear to what extent the currently deployed CGN base complies with the aforementioned NATs requirements. There is also the concern that CGNs utilization may result in perfor-

[23] M. Ford, M. Boucadair, A. Durand, P. Levis, and P. Roberts. Issues with IP Address Sharing. RFC 6269, June 2011

[24] C. Donley, L. Howard, V. Kuarsingh, J. Berg, and J. Doshi. Assessing the Impact of Carrier-Grade NAT on Network Applications. RFC 7021, September 2013

[25] Z. Wang, Z. Qian, Q. Xu, Z. Mao, and M. Zhang. An Untold Story of Middleboxes in Cellular Networks. In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, pages 374–385, 2011. ISBN 978-1-4503-0797-0; and S. Triukose, S. Ardon, A. Mahanti, and A. Seth. Geolocating IP Addresses in Cellular Data Networks. In *Passive and Active Measurement*, pages 158–167. Springer, 2012

mance penalties. It is a common operational practice that a single CGN serves a very large number of mobile nodes (up to millions of devices per CGN). Because of this, the CGN itself may become a bottleneck and impose performance penalties to the communication. This is especially so for the case of the first packet that creates the NAT mapping, but it may also affect traffic forwarding once the mapping has been established. Measuring and understanding the performance penalties resulting from the use of CGNs is important for application designers, for end users and for mobile operators adopting the CGN technology. (e.g. for application designers, this information should affect the maximum number of communications open in parallel). This input is very relevant for a measurement platform such as MONROE, since the performance penalties if exist can affect and bias the measurement results.

In order to achieve a more deterministic behavior from the NAT boxes, the Internet Engineering Task Force (IETF) produced a number of specifications defining the requirements that NATs should follow when creating, preserving and removing their internal state as well as some recommendations in terms of the different filtering and forwarding policies that NAT should implement. In particular, the IETF released NAT behavioral requirements for handling TCP traffic [26], UDP traffic [27] and ICMP packets. [28] However, since these IETF standards specify the internal behavior for a NAT, it is far from trivial to assess whether NATs are following the recommendations and to the best of our knowledge there is no information about the prevalence of NAT boxes that honor the IETF specifications.

The contribution of this part is therefore three-fold:

- First, we design and develop NATwatcher, a tool to measure key aspects of the behavior of the deployed NAT base, along with a measurement methodology based on crowdsourcing that allows us to perform large scale measurement using NATwatcher. We perform a large measurement campaign and we deploy NATwatcher in over 700 measurement points, building a large data set describing the behavior of over 700 NAT boxes from 65 different countries and 280 ISPs, testing over 120 different NAT vendors. We find that a large majority (80%) of the NAT boxes we tested follow the IETF recommendations for 11 out the 17 of the considered features. We also observed that about half of the NAT devices we tested exhibit the exact same behavior for all the features we tested. While the other half of the devices use a variety of configurations.

- Second, we perform a large scale active measurement campaign to detect CGNs in fixed broadband networks using NAT Revelio. [29] Revelio enables us to actively determine from within residen-

[26] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. IETF, RFC 5382, October 2008

[27] F. Audet and C. Jennings. Network Address Translation (NAT) Behavioral Requirements for Unicast UDP. IETF, RFC 4787, January 2007

[28] P. Srisuresh, B. Ford, S. Sivakumar, and S. Guha. NAT Behavioral Requirements for ICMP. IETF, RFC 5508, April 2009

[29] A. Lutu, M. Bagnulo, A. Dhamdhere, and K. Claffy. NAT Revelio: Detecting NAT444 in the ISP. In *International Conference on Passive and Active Network Measurement*, pages 149–161. Springer, 2016

tial networks the type of upstream network address translation, namely NAT at the home gateway (customer-grade NAT) or NAT in the ISP (Carrier Grade NAT). We deployed Revelio on two large-scale hardware-based measurement platforms – RIPE Atlas in Europe and the FCC "Measuring Broadband America" (FCC-MBA) in the U.S. – with a total of 5,121 vantage points in over 60 ISPs. The FCC-MBA deployment consisted of 2,477 home routers operated by SamKnows in 21 large residential broadband Internet access service providers in the U.S. We also adapted the Revelio methodology to run on the RIPE Atlas infrastructure (using their available user tests), and executed the tests from 2,644 Atlas probes in 43 ISPs mainly active in Europe. We thus demonstrate the flexibility of Revelio and exemplify the use of two fundamentally different large-scale measurement testbeds.

Our results show that 10% (6 out of 64) of the ISPs we tested have some form of CGN deployment. In particular, one ISP has a large-scale deployment where Revelio detected upstream CGN deployment from all 76 vantage points in that ISP. In the other 5 ISPs we observed evidence of a localized deployment limited to a subset of customers. We verified our results with representatives of the ISPs to validate our positive and negative inferences at the IP level. We confirmed the results for 4 of the 6 positive ISPs by personal communications with ISP representatives. The combination of the FCC-MBA and RIPE Atlas study represents (to the best of our knowledge) the largest active measurement study to date with confirmed CGN deployments in broadband networks at the IP-level granularity.

- Finally, we create and test CGNwatcher. The goal of CGNwatcher is to design, develop, implement, deploy an extensive set of metrics to characterize CGN behaviour, to provide useful insight about the current CGN base deployed in the Internet.

As part of the thesis, we characterize roaming across Europe for 16 different MNOs deployed in six countries. The goal of the experiment is to try to understand if the MNO changes CGN while roaming, for this reason, we run a series of measurements that enable us to identify the roaming setup, infer the network configuration for the 16 MNOs that we measure and quantify the end-user performance for the roaming configurations which we detect.

International roaming allows mobile users to use their voice and data services when they are abroad. The European Commission (EC), in an effort to create a single digital market across the European Union (EU), has recently (as of June 2017) introduced a set of regulatory decisions [30] as part of the "Roam like Home" initiative. This

[30] European Commission: New Rules on Roaming Charges and Open Internet. https://ec.europa.eu/digital-single-market/en/news/new-rules-roaming-charges-and-open-internet. [Online; accessed 06-March-2018]

initiative abolishes additional charges for users when they use voice and data services while roaming in EU.

In this setting, Mobile Network Operator (MNO) are expected to deliver services with Quality of Service (QoS) properties similar to the ones a user experiences when at home.

Several topology architectures can be used for roaming in a mobile network, namely, home-routed roaming (HR), local breakout (LBO) and IPX hub breakout (IHBO).

The topology can have a potential impact on the communication performance. For instance, when the node accesses services inside the *visited network*, the performance is likely to be worse in the HR case, because all packets travel twice between the visited and the home country; less so when the communication peer is in a third country and is minimal when accessing services in the home country.

In the last part of this work, we perform an extensive large-scale measurement study to understand the roaming ecosystem in Europe after the "Roam like Home" initiative. More specifically, we investigate:

- Which technical solutions are actually being deployed and used today?
- What are the implications of roaming on the service experienced by the roaming user?

To address these questions, we built a unique measurement platform, *MONROE-Roaming*, to assess roaming and its performance implications. The platform integrates dedicated measurement hardware that we deployed in six different countries across Europe, covering a total of 16 MNOs. We purchased Subscriber Identity Modules (SIMs) that support roaming for these MNOs and distribute them across the six countries. We characterize roaming operation and network performance and evaluate the impact on VoIP and web applications while roaming. We find that all observed MNOs use HR, which yields noticeable latency increases. We do not observe traffic differentiation policies for VoIP or web, but we do find evidence of content discrimination for roaming users.

2 *State of the art*

A BROAD RESEARCH CORPUS is available when studying the deployment of a new protocol in the Internet. Here we select a subset of works that share some similarities with our work Honda et al. [2011], Hirth et al. [2015]. Honda et al. Honda et al. [2011] shed some light on the feasibility of extending TCP due to its (bad) interaction with middleboxes. Authors demonstrate that middleboxes are almost omnipresent and that they can arbitrarily change packets header (e.g., dropping both known and unknown TCP options) or payloads, particularly over port 80.

The extent to which the Internet deploys and supports ECN has been a topic visited repeatedly over the past 15 years Medina et al. [2005], Langley [2008], Bauer et al. [2011], Trammell et al. [2015], Bhooma [2017]. Previous work focused on three main aspects of ECN-readiness, namely, server-side support, end-to-end path support and, finally, client-side support.

Measurements of server-side support usually focus on the population of web servers as ranked by Alexa. While quantifying ECN server-side support in 2004, Medina et al Medina et al. [2005] found that as little as 2.1% of the servers tested supported ECN. Ten years later, Trammell et al Trammell et al. [2015] report an acceleration in the deployment of ECN-capable servers, finding 56.17% of ECN-capable servers.

In this work, we corroborate the acceleration of ECN deployment for wired networks and Alexa servers, and we extend the study to mobile networks and ECN++.

Regarding support of ECN by the network elements on the end-to-end path, in 2013, Kühlewind et al Kühlewind et al. [2013] tested 22,487 hosts and reported that in 0.9% of cases ECN was not usable due to middleboxes along the path. Later, in 2015, Trammell et al Trammell et al. [2015] finds that when testing 326,743 hosts capable of negotiating ECN, for 0.02% of them (107 hosts) a device on the path mangles the TCP/ECN flags. We note that all these above-mentioned efforts report on measurements performed mostly in fixed networks.

Only recently, in March 2017, Apple Bhooma [2017] reported 100% positive results after testing from vantage points connected to 'a few' mobile carriers, which on further investigation meant three carriers Cheshire [2016].

As mobile broadband networks accommodate more traffic, there is a pressing need for similar analysis of mobile carriers, which are known to represent a middlebox-rich environment. Because of the difficulty of instrumenting mobile devices to perform ECN measurements, there is little prior work Kühlewind et al. [2013], Bhooma [2017]. For example inHonda et al. [2011, opt] the authors only have access to a low number of vantage points. To overcome the latter issue, Hirth et al. Hirth et al. [2015] demonstrate that crowdsourcing platforms can become a powerful tool to achieve a realistic view of the network from an end-user perspective.

In this work, we present the largest measurement study of ECN support in mobile networks to date, using vantage points connected to 18 mobile operators. The results we find in this work contrast with the prior work, providing a different picture of the current support of ECN (in particular in mobile networks) and unraveling significant challenges yet to overcome towards global ECN adoption.

As we described earlier, we also propose to design and implement a set of tests that measure a number of functional and performance metrics to characterize NATs and particularly CGNs. Due to the difficulties that are inherent to perform large-scale measurements in real residential environments, to date, the few studies that are available have performed testing of different NAT devices in a lab environment. In Hätönen et al. [2010] authors study the configurations of 34 different home gateway models, analyzing the processing of various TCP and IP options and measuring the success of some network protocols when traversing NATs (i.e., STUN Rosenberg et al. [2008a], TURN Mahy et al. [2010] and ICE Rosenberg [2010]). In Jennings [2007] authors perform a lab study of the support of a number of features such as mapping, filtering, hairpinning, on 42 NAT device models. While these studies provide some information about the capabilities of the different NAT boxes, they provide limited information about the actual behavior of the deployed NAT devices in the Internet. This is so because the behavior of the deployed NAT base also largely depends on the configuration of the NAT boxes and in the popularity of the different NAT products.

The usage and impact of CGN-based solutions Skoberne et al. [2014] has lately drawn much attention from the community Donley et al. [2013], Ford et al. [2011]. Consequently, we have seen several approaches for quantifying and confirming the degree in which operators are actively deploying operational CGNs Müller et al. [2013], Richter et al. [2016].

Richter et al. Richter et al. [2016] use passive measurements to quantify CGN deployment rate in the Internet, after observing that some nodes in the BitTorrent DHT mistake addresses internal to a CGN for external addresses and therefore propagate these IP addresses to other nodes. The authors report the average CGN penetration rate to be 17-18% of all Eyeball ASes. In this work, however, we focus on active CGN detection in fixed-line broadband providers at the IP level. NAT Revelio empowers end-users to test whether their upstream provider connects them behind a CGN solution active in the access network. Similar to our efforts, the Netalyzr Kreibich et al. [2010] tool, initially meant as a networking debugging tool, has been repurposed to detect CGN solutions in mobile networks. The authors present the measurements results in Richter et al. [2016], which corroborate the conclusion of prior work Triukose et al. [2012], Wang et al. [2011] that for mobile providers CGN deployments are commonplace. For wireline networks, however, it is non-trivial to detect and confirm CGN deployment on a per-IP basis.

Another notable effort towards passively quantifying the degree of CGN deployment is Livadariu et al. [2017]. The authors look at CGN deployment with a /24 IP prefix granularity. For the 92 lines Revelio identified as being behind a CGN, we identified 52 covering /24 prefixes and cross-compared our results with Livadariu et al. [2017]. We find that 38 prefixes appear as CGN-positive in Livadariu et al. [2017], while 13 prefixes were classified as not being used in CGN deployments and one was unaccounted. Given that we were able to validate at the IP level 89 of these IPs, we thus highlight the benefits of an active measurements approach, while acknowledging the advantage of the passive approach to easily scale.

Finally, using an earlier version of NAT Revelio, the authors presented in Lutu et al. [2016] a smaller study on CGN deployment in the United Kingdom showing a very low penetration degree. This work presents an evolution of the Revelio methodology that includes additional tests designed to support the myriad of technologies and devices deployed in real operational environments, as described in Section 8.2.

In terms of measuring the performance of CGNs, there is very little work available. In particular in Bocchi et al. [2016] authors measure the RTT, the time required for the TCP 3-way handshake, the time to start the data transfer and the goodput rate. While, in Ohara et al. [2014] authors analyze the impact of network delays when a CGN is used. We believe that there are other relevant metrics that are useful to understand the potential limitations introduced by the CGN. For instance, the performance of the CGN when creating new mappings (e.g. maximum rate at which mappings are created, and whether this affects the forwarding of packets for which a mapping already exists) or metrics regarding the performance of the CGN when the number of ports allocated to a given end user increases.

International roaming has received little coverage in terms of large measurement studies, potentially because of the high costs and coordination efforts associated with running such a campaign. Vallina *et al.* Vallina-Rodriguez et al. [2015b] has leveraged crowdsourced measurements and focused only on national roaming agreements between MNOs in France. The study does not provide any further evaluation in terms of performance or content implications. Using controlled measurements in the dedicated platform MONROE Alay et al. [2017b] enabled Michelinakis *et al.* Michelinakis et al. [2018] to analyze the impact of international roaming, but only for two operators in Europe. They find that the home-routed configuration does impact the performance of cloud service providers, such as Akamai or CloudFront. Our work complements this work and presents an extensive measurement study to understand the international roaming ecosystem in Europe since the "Roam like Home" initiative.

There have been myriad recent studies focusing on mobile network characterization and performance. For instance, while Huang *et al.* Huang et al. [2013] study LTE network characteristics in a cellular operator in the US, Safari *et al.* Khatouni et al. [2017] show performance measurement in mobile networks are much more complex than wired networks, due to the different network configurations such as the presence of Network Address Translation (NAT) or Performance Enhancing Proxies (PEP), which do vary over time. Kaup *et al.* Kaup et al. [2016] run a crowdsourcing campaign to measure RTT and throughput towards popular websites in Germany. They used the dataset to show that the association of a mobile endpoint to the Point of Presence (PoP) within the operator network has influence on network performance. The authors of Molavi Kakhki et al. [2015] present a mobile app and a mechanism for identifying traffic differentiation for arbitrary applications in the mobile networks. Ververis *et al.* Ververis et al. [2015] surveys content filtering for a mixture of broadband and cellular ISPs and finds a lack of transparency around the policies they implement as well as outdated and poorly implemented blacklists. In our work, we not only focus on network performance of roaming infrastructure, but also identify possible traffic differentiation for particular applications and content discrimination and geo-restriction for users in international roaming.

2.1 *Existing Crowdsourcing Large Scale Measurement Platforms*

In this Section we describe in details the existing large scale measurement platforms. in the next Chapters we use those platforms to make informed protocol design choices.

2.1.1 *RIPE Atlas*

RIPE Atlas [1] is an active Internet measurement network from the RIPE NCC. It consists of thousands of measurement probes distributed around the globe (see Figure 2.1).

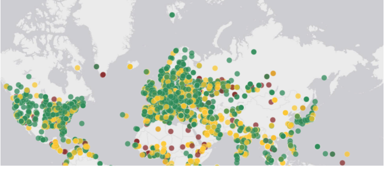


Figure 2.1: RIPE Atlas network map.

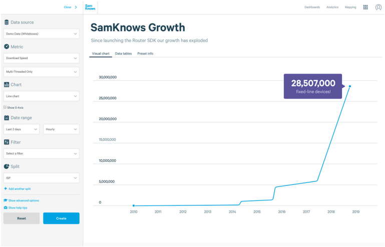
A probe (figure 2.2) is a small hardware, located in people's home, that runs measurements in the RIPE Atlas system and reports these results to the data collection components. A host is someone who hosts a probe for RIPE Atlas. A RIPE Atlas host earns credits for the time that his probe is connected. It is possible to use these credits to conduct user-defined measurements, such as Ping, Traceroute, DNS, SSL and HTTP, using the entire RIPE Atlas network.

Figure 2.2: RIPE Atlas network.

2.1.2 *SamKnows*

SamKnows [2] is a measurement platform with installed whiteboxes in over 28,507,000 homes, across many different ISPs. The platform was born for measuring performance of access networks in UK and in few year it expanded around the world (Figure 2.3). The gateway is a Netgear WNR3500L RangeMax Wireless-N Gigabit router with a 480 MHz MIPS processor, 8 MB of flash storage, and 64 MB of RAM.

Samknows can be used by regulators, ISPs and manufactures.



SamKnows Growth

Since launching the Router SDK our growth has exploded

Data source: Demo Data (Whiteboxes)

Metric: Download Speed

Multi-Threaded Only

Chart: Line chart

□ Show 0-Axis

Date range: Last 2 days, Hourly

Filter: Select a filter

Split: ISP

28,507,000 fixed-line devices!

Figure 2.3: SamKnows growth.

2.1.3 *MONROE*

MONROE Alay et al. [2017a] is the first open and flexible platform to run experiments on operational 3G/4G Mobile Broadband networks. It is composed by 450 MONROE nodes, distributed across Norway, Sweden, Italy, Spain, Greece and UK. 150 MONROE nodes are devoted to mobility and installed on trains and buses. MONROE provided also a scheduler web interface, making easy to access the system and deploy experiments on all or a selected subset of the nodes. The platform provides also metadata, context information about the node useful during the process of the results from the experiment.

Figure 2.4 shows the system design of MONROE. The nodes are located in bus, trains or people's home and each connect to two or three MNOs. The scheduler allows the access to external users, assigning a given slot of time. The experiments results are pulled to

[1] https://atlas.ripe.net/

[2] http://www.samknows.com

external repositories, accessible later by the users.

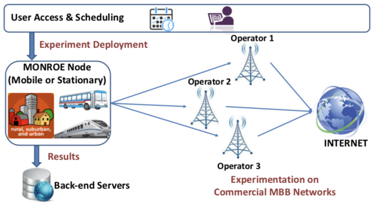


Figure 2.4: The MONROE platform.

Part I Feasibility of New Protocols Extensions

3 *Leveraging Crowdsourcing Platforms for Network Measurements*

By expanding the traditional crowdsourcing focus from the human element to use a diverse and numerous group of end-user devices as measurement vantage points, we show how to leverage on crowdsourcing platforms and how to exploit measurements platforms to run Internet wide measurements.

3.1 Microworkers Crowdsourcing Platform

The Internet is nowadays the main venue for crowdsourcing, since anyone with an Internet connection can become involved. Crowdsourcing platforms connect *employers* and *workers* from around the world.

The employer is the one who creates the task (or the *"micro-job"*) for workers and specifies the parameters of its campaign, e.g., the size of the set of users performing the task or their geographical location at country level. Each worker meeting the required criteria can carry out the task only once, thus validating the uniqueness of the results for which the employers pays. The platform then acts as a referee, guaranteeing to the employer that the campaign funded is successfully completed within the required parameters, while also making sure that the workers involved get paid for their contribution.

Microworkers offers world-wide access to employers, unlike similar more popular crowdsourcing platforms[1]. Furthermore, it offers an automatic payment method based on a unique verification code, called VCODE. The latter provides the worker a proof (payment code) for each task performed on the external page, so that the payment can be handled via the Microworkers platform. The platform presents a simple interface and it is very reliable, allowing an employer to refuse pay for work which does not meet the required quality. Another important advantage of using Microworkers for Internet measurements is the possibility to select the vantage points based

[1] For example, Amazon Mechanical Turks is only available to employers based in the U.S., thus restricting our particular access.

on certain criteria, such as the geographical location at the country level, the type of Internet access (fixed or mobile) or even the type of measurement equipment used to perform the tasks (i.e., Android or iOS mobile operating system).

3.2 *Informing Protocol Design through Crowdsourcing*

Modern networks often rely on dedicated hardware components generically dubbed middleboxes to perform advanced processing functions like, for example, enhancing application performance (e.g., traffic accelerators, caches, proxies), traffic shaping (e.g., load balancers), optimizing the usage of IPv4 address space (e.g., NATs) or security (e.g., firewalls).

One major criticism of middleboxes is that they might filter traffic that does not conform to expected behaviors, thus ossifying the Internet and rendering it as a hostile environment for innovation. [2] Middleboxes for example can alter HTTP requests and responses, especially headers, to provide their functionalities. [3] The unpredictable and diverse behaviors of such middleboxes can be problematic when adopting new protocols. This is a well know problem, and an IRTF research group is setup [4] to quantify its extent and propose solutions.

This does not mean that it is impossible to deploy new protocols, but that in order to ensure success it is imperative to first understand the interaction of the proposed solutions with the middleboxes active along the path. Recent studies on middleboxes behaviour attempt to provide such information. In order to perform representative Internet measurements and test realistic scenarios and different Autonomous Systems (ASes), what is missing is access to a high number of diverse vantage points. In addition, more evidences of the nature of the problems and the portions of the network in which they manifest are needed.

Figure 3.1: Some examples of middleboxes.

Crowdsourcing platforms present a simple interface and they are very reliable. Another important advantage of using crowdsourcing platforms for Internet measurements is the possibility to select the vantage points based on certain criteria, such as the geographical location at the country level, the type of Internet access (fixed or mobile) or even the type of measurement equipment used to perform the tasks (i.e., Android or iOS mobile operating system). However, some information cannot be extract directly from the platform, so that the interaction with the users appears essential to obtain some relevant data to perform a realistic analysis. To overcome this issue, we create a survey in which we ask to the users what kind of connection they are using and other information that help us to have a complete view of the Internet scenario. We combine the crowd-

[2] M. Handley. Why the internet only just works. *BT Technology Journal*, 2006

[3] N. Vallina-Rodriguez, S. Sundaresan, C. Kreibich, and V. Paxson. Header Enrichment or ISP Enrichment?: Emerging Privacy Threats in Mobile Networks. In *Proceedings of the 2015 ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, pages 25–30. ACM, 2015a

[4] https://www.ietf.org/mailman/listinfo/hops. *IRTF*, 2015

sourcing solution approach with the access to one of the available crowdsourcing large-scale measurements platforms, like for example the RIPE Atlas platform, SamKnows, PlanetLab and MONROE.

The MONROE platform is also ideal for our purposes due to its flexibility allowing the deployment of all kind of sophisticated tests that include the crafting of specific packet sequences, thing quite difficult on crowdsourcing platform, since the user does not own a rooted device. MONROE provides a somehow controlled environment, as the hardware running the tests is under control of the MONROE consortium. Being part of the external user, MONROE allows us to complement the main advantage of crowdsourcing techniques that is scalability with repeatability and control of the measurement process. Mostly due to privacy reasons, crowdsourcing platforms measurements do not always provide important context information like location, kind of equipment, and connection status. Moreover, MONROE supports the deployment of different applications and protocols, and enables benchmarking tools and methodologies.

microWorkers
work & earn or offer a micro job

Existing user Login New user? Register for free

Employers,
post a micro job

Employers, ask people to...

— Blog about your product
— Post reviews to Websites & Blogs

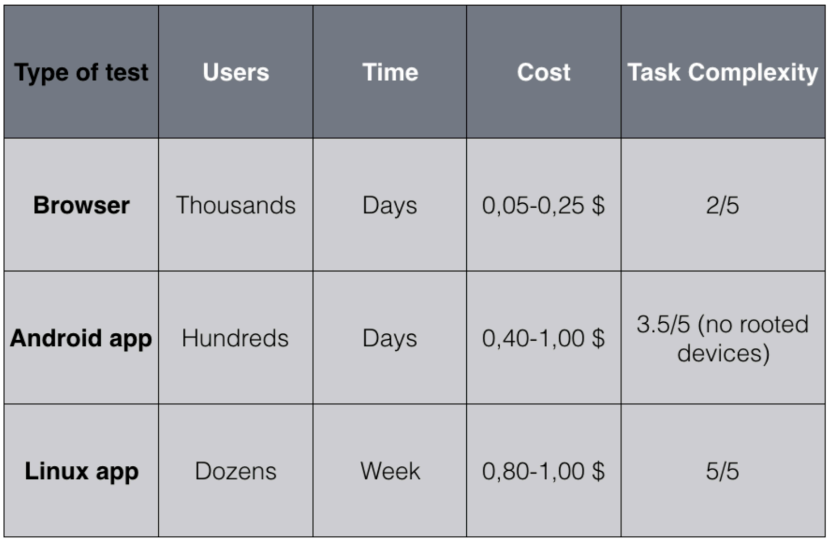
Workers,
get paid to do micro jobs

Workers, sign up and...

— Browse micro jobs
— Select jobs you like

Figure 3.2: Microworkers crowdsourcing platform.

3.3 Experiences and lessons learned measuring the Internet with crowdsourcing



Type of test	Users	Time	Cost	Task Complexity
Browser	Thousands	Days	0,05-0,25 $	2/5
Android app	Hundreds	Days	0,40-1,00 $	3.5/5 (no rooted devices)
Linux app	Dozens	Week	0,80-1,00 $	5/5

Figure 3.3: Crowdsourcing Platform Guidelines.

The experience on studying the seven case studies described before gave us the opportunity to understand the functionalities of crowdsourcing platforms for network measurements. Figure 3.3 shows the type of tests we offer to the crowdsourcing platform users. The browser test is the simpler and the most common and cheap.

Since what users need is only a standard browser in few days is possible to get thousands of vantage points. It is possible to deploy an Android or Apple app and serve through the crowdsourcing platform. In this case the users need to own a smartphone and in few days it is still possible to have hundreds of users, but with a higher cost.

Finally, the linux app is an application that users need to download and run on a linux computer. Since Windows OS is more common between users, it is hard to recruit users with such equipment and the cost can increase. Despite this, we could recruit dozens of users in few weeks.

Testing crowdsourcing platform for network measurements, we understood that the best period for launching a campaign is during weekend. We recruited more users than during the week.

Creating survey for understanding users network context can be hard and users can lie. We deploy some methodologies for understanding users behavior and the fairness of their answers. Like for example changing the order of the questions. We discovered that 60% of the users lie. We did not use that context information in that case.

4 The Feasibility of Pervasive Encryption

In this chapter, we show how to make informed protocol design choices by using a novel methodology for performing large scale Internet measurements, using a crowdsourcing solution approach. We exemplify next the efficiency of our methodology in the case of evaluating the feasibility of pervasive encryption in the modern Internet ecosystem.

4.1 The Feasibility of Pervasive Encryption in the Internet Ecosystem

The long-term objective of the research community is to provide encryption by default for all Internet communications. In order to achieve that, the use of TLS as a substratum for all communications is being considered. In particular, the IETF *tcpinc* working group devoted to provide ubiquitous, transparent security for TCP connections considering the use of TLS.

The IETF also longly discussed about whether HTTP/2 should be encrypted by default using Transport Layer Security (TLS). In the one hand, encryption provides strong privacy guarantees for the end-users and also reliability, encrypted streams in fact, have higher success probability to traverse the network without modification. In the other hand, the use of encryption has some harmful implications:

- It imposes some server overhead and can degrade user performance in high-latency links, like satellite ones [1] or add extra CPU load that small devices cannot handle.

- There is a requirement to inspect or cache HTTP traffic for a multitude of reasons, (3) certificates are too expensive.

Based on all these considerations, the final HTTP/2 standard defines two way of supporting HTTP/2: HTTP/2 over TLS (H2) and HTTP/2 in the clear (H2C), or directly over TCP. Specifically, the standard defines that H2C should be negotiated by mean of an Upgrade header field in a classic HTTP/1 request.

[1] D. Naylor, A. Finamore, I. Leontiadis, Y. Grunenberger, M. Mellia, M. Munafò, K. Papagiannaki, and P. Steenkiste. The Cost of the S in HTTPS. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 133–140. ACM, 2014a

Despite major browser vendors currently only support H2, several websites already support H2C. As of December 2015, 30,000 websites announce support for H2C though only about 80 sites fully support the header Upgrade mechanism. [2] Motivated by the above observations, in this work we set out to quantify the feasibility of H2C in today's Internet.

However, before designing any solution, it is first essential to understand the feasibility of pervasive encryption in the Internet ecosystem by measuring the interaction of middleboxes with the TLS across the different TCP ports that currently use plain text protocols.

In other words, we need to establish at this point whether using encryption in traditionally unsecured ports is even possible in today's Internet. In this work, we attempt to initiate TLS and H2C connections in 68 different ports that normally do not use any form of encryption and analyze the success of the connection. This is a first necessary step towards a full comprehension of the behavior of middleboxes relative to pervasive encryption.

4.2 *HTTP/2: Operations and Characteristics*

HTTP/2 aims to address a number of issues of HTTP/1 by providing an optimized transport of HTTP semantics in a server-client connection.

HTTP/1 does not really work as a full duplex connection, most of the time the connection is in an idle state as only one request at a time can be placed in a connection. Moreover HTTP/1 is very verbose and take a considerable number of bytes to transmit, mainly because of Cookies, thus slowing down initial requests and adding a lot of overhead sending the same bytes in every request. Another problem is that HTTP/1 is a text protocol, so it appears hard for machines to parse it. Text protocols are not efficient neither easy to implement correctly: optional white spaces, different termination tokens and other quirks make it harder to differentiate between the protocol and the payload.

In order to solve all of these issues, HTTP/2 introduces a **binary** framing layer. Requests and responses are broken down into multiple frames and transferred between client and server in the same connection. Each tuple request-response runs in its own stream, allowing the multiplexing of several requests and responses in a single TCP connection by interleaving frames. These changes allow new features like flow control and prioritization. Another benefit of using a single connection is that the server can push data to the client and send resources even before the client knows they will be needed. Due to all these changes the HTTP/2 frame format is, indeed, completely

[2] Is the web HTTP/2 yet? *http://isthewebhttp2yet.com/*, 2015b

different from the HTTP/1.

The high-level API of the HTTP protocol remains exactly the same and the changes are only low-level, to address the performance limitation of the protocol and add extra features. There are two groups of frames:

- HEADERS, PUSH_PROMISE, CONTINUATION and DATA which are used mainly to send data between peers and they are associate with a stream;
- PRIORITY, RST_STREAM, SETTINGS, PING, GOAWAY and WINDOW_UPDATE frames are used for flow-control, management and configuration in the connection and streams.

All frames are composed by a 9-octet header and a variable-length payload as shown in Figure 4.1.

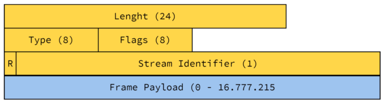


Figure 4.1: HTTP/2 frame layout.

Once the connection has been established, either plain TCP using HTTP upgrade mechanism or via TLS, both ends send a connection preface as final confirmation of HTTP/2 being used. Client and server send a SETTINGS frame to indicate the configuration parameters which are not negotiated and must be acknowledged by the other end. Then the client can start sending HTTP requests to the server. First the client sends a HEADERS frame, followed by some CONTINUATION frames. As HTTP headers are stateful for the connection in HTTP/2, to avoid race-conditions, no other frames can be sent until they are sent. If there is some body in the request, DATA frames are used. The same process will be used by the server to respond. This happens concurrently for multiple communications on the same connection.

If server push is enabled, the server can send a PUSH_PROMISE frame for data that it will be requested by the client. This frame is basically a request sent by the server instead of the client. Once the PUSH_PROMISE frame is sent, the server can send DATA frames with the response. As PUSH_PROMISE must be cacheable, this streams can be closed by the client if the resource is not needed.

In contrast with HTTP/1, HTTP/2 uses one connection for multiple requests-responses. Once the connection is not longer needed, a GOAWAY frame is sent to terminate gracefully the connection. If an issue occurs during the connection and it becomes unusable, client and/or server can close the TCP connection. They should first try to send a GOAWAY frame.

HTTP/2 uses the same HTTP and HTTPS URI schemes used by HTTP/1 and it shares the same default ports: 80 for HTTP URIs and 443 for HTTPS URIs. HTTP/2 standard offers two kinds of implementation:

- over TLS (H2), using the ALPN or NPN identifiers for protocol negotiation;

- over TCP (H2C), in which the initial Upgrade request must be implemented. In this case the client sends an HTTP/1 request that includes an Upgrade header field with the H2C token. The server accepts the Upgrade and responds using a 101 (Switching Protocols) status code, showing that the server is changing to a different version of HTTP.

These frames must include a response to the request that initiated the upgrade.

In this work we test the two implementations of the protocol (i.e., over TLS and using Upgrade), additionally we test H2 without ALPN negotiation and H2C without the Upgrade request. A client, that makes a request for an HTTP URI without prior knowledge about support for HTTP/2, uses the HTTP *Upgrade mechanism*. Basically, the client makes an HTTP/1 request that includes an Upgrade header field with the H2C token. A server that supports HTTP/2 accepts the Upgrade with a 101 (Switching Protocols) response. After the empty line that terminates the 101 response, the server can start sending HTTP/2 frames. These frames must include a response to the request that initiated the upgrade.

4.2.1 HTTP/2 Encryption

A client that makes a request to a HTTPS URI uses TLS with the application-layer protocol negotiation (ALPN) extension. The protocol identifier in this case is HTTP/2. Once TLS negotiation is completed, both the client and the server must send a connection preface (Figure 4.2).

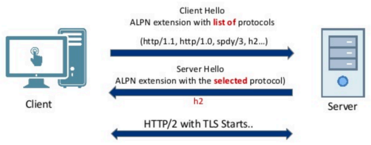

Client Hello
ALPN extension with list of protocols
(http/1.1, http/1.0, spdy/3, h2...)
Server Hello
ALPN extension with the selected protocol)
h2
HTTP/2 with TLS Starts..
Client
Server

Figure 4.2: HTTP/2 ALPN extension.

4.3 The Case of the Pervasive Encryption: Experimental Setup

We try to establish TLS connections from a large number of vantage points (from now on, *measurement agents (MAs)*) to a large number of ports, which traditionally do not use TLS in a target server (from now on, *measurement server (MS)*), using crowdsourcing based measurements.

We realize our measurement campaign—to study TLS HTTP/2 interaction with potential middleboxes—through Microworkers crowdsourcing platform. In order to attract a large set of workers, our "microjob" should be as simple as possible, which is quite challenging when performing Internet measurements. [3] Workers should not be required to have any technical background, and all data should be

[3] V. Paxson. Strategies for sound internet measurement. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 263–271. ACM, 2004

automatically collected and reported to us, without any worker input. Ideally, workers should perform only a simple task like visiting a Web page from a browser, or launching a (provided) application on their mobile phone.

4.3.1 TLS Campaign Setup

Measurement Server. To capture how effective would pervasive encryption actually be if deployed in today's Internet, we collect and analyze the results from more than 2,000 MAs that try to establish TLS connections in a large number of ports which normally do not use TLS. The target of our tests is a dedicated server (MS) which we are able to control and which receives the communication from the different measurement agents.

The MAs attempt to establish both HTTP and TLS connections to 68 different ports, namely 10 well-known ports, 56 registered ports and 2 ephemeral ports. All the server ports we select are ports that normally use HTTP-based protocols, such that the HTTP connection is effectively what any middlebox would be expecting in that port. We select only 2 ephemeral ports, since we expect that the behavior for the rest of ephemeral ports is similar. We use the success rate of the HTTP connections as the benchmark against which we compare the number of TLS successful connections. We establish then the success rate of the TLS connection by contrasting the result against the status of an unencrypted HTTP connection established in the same port.

We have installed two dedicated servers to collect clients' HTTP and HTTPS activity. Information about the attempted HTTP and HTTPS connections are stored for the post-processing phase. We utilize the LAMP model (Linux, Apache Server, MySQL relational database management system, PHP) to allow storage and retrieval of data in a modifiable format using simple query APIs. We also store and analyze in detail the server side packet exchanges.

Measurement Agents. When designing our experiment, we want to capture and compare the results from MAs that include both fixed and mobile Internet access. To this end, we create several international crowdsourcing campaigns directed either to mobile user or desktop users exclusively. We recruit in total 2,120 MAs. We specify the number of users we want to recruit, the Country in which we want to direct the campaign an the kind of campaign we want to spread. However, when designing Internet measurements crowdsourcing campaigns, one has to bear in mind the fact that the level of direct control over the end-user and the measurement device is still limited. Some information that is of importance for the designed experiment may not be available through the platform. For this reason we create a methodology to collect the data from each MA completing the task. The procedure is as follows. First, we start by asking the user to connect using a HTTP connection in port 80 to a webpage we provide. Meanwhile, in the background, HTTP and HTTPS connections are performed from the measurement devices to our servers in all the other 67 ports. In this case, data about the performance are collected in the MS.

Second, the webpage we provide contains a short form asking for additional input about the type of Internet access they are using. This allows us to gather the information that the crowdsourcing platform cannot provide directly and that we are not able to collect on the server side. The fixed-line MAs can indicate the place from where they are connecting (*Home*, *Hot Spot*, *University or other institution*, *Company*). Additionally, for the category *Home*, we offer 4 different choices for the residential access technology used, namely *Cable*, *xDSL*, *Fiber* and *Other*. Contrariwise, the workers in the campaigns for mobile access can choose between *2G*, *3G* or *4G*, based on the mobile technology they are using.

Finally, on the server side, we also collect and store metadata on each of the MAs that connect to our servers, such as the IP address, the user agent type, the language, and any other information included in the HTTP header.

4.3.2 *HTTP/2 Campaign Setup*

Measurement Tool. We build the tool with one focus in mind: simplicity for the test workers. The workers do not necessarily have any technical background and, as external entities to the test, all the information should be produced and collected within our tool, with minimal requirements and input from them. We built two clients applications (a Web application and an Android app) that connect to two Web servers (for non secure and secure communications). The reason for which we use two different clients is that HTTP/2 negotiation is done during the handshake using the extension ALPN, this extension has limited support in Android because of a bug. [4] For this reason, we modify the Android library to remove the ALPN negotiation and perform HTTP/2 connections over TLS without ALPN negotiation. To perform the test using HTTP/2 over TLS with ALPN negotiation, we use the workers browser (i.e., Chrome, Firefox and Opera) that currently support HTTP/2 only over TLS. On the server, we built a PHP application that provides the information to the clients and saves the data generated by them, storing it in a

[4] https://github.com/square/okhttp/issues/1305. *Don't use broken ALPN on Android 4.4 | GitHub*, 2015

MySQL database for post-processing phase.

We create two different crowdsourcing campaigns, a browser campaign in which workers, using their own browser, need to follow a link to the application that performs h2 connections with ALPN negotiation in the 68 ports and an Android campaign in which workers need to download an app that performs h2 connections without ALPN negotiation, h2c and h2c without Upgrade connections in 68 different ports, first using mobile network and then if available the WiFi connection to test fixed line. The procedure to collect measurements is as follows. Following the information in the Microworkers platform, workers are redirected to our web pages containing the instruction to perform the tasks. Basically they need only to press the button *Start*, then the application is responsible for performing all the tests.

Our tool consists of a set of resources that can be accessed via URI endpoints based on a REST architecture, these resources can be invoked using JSON inputs. We implement 3 endpoints in the server: Start, Test and Finish. Workers make a request to the Start endpoint to obtain a list of URLs to perform the test on. These URLs point to the Test endpoint with different combination of hosts and ports. Once the test is completed, the application confirms it by making a finally request to the Finish endpoint.

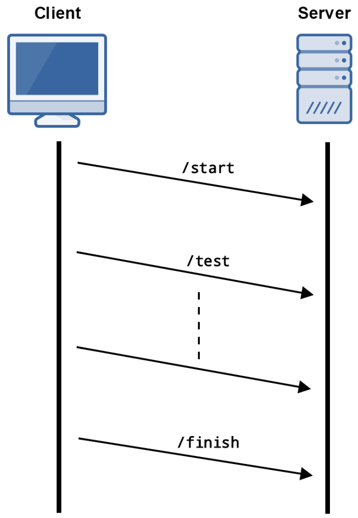


Figure 4.3: Server Endpoints.

The server endpoints (Figure 4.3) provide the following functionality:

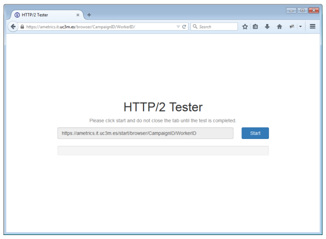
- *Start*: a message with an unique token and a list of URLs to perform requests on is returned. The token helps to distinguish between different requests produced by the workers, even by the same worker.
- *Test*: this endpoint receives the token as part of the URL. HTTP1.1, h2, h2 without ALPN, h2c and h2c without Upgrade requests are performed from the workers to our servers over 68 different ports. If the request is correct, a JSON message confirms it and also returns the protocol used. This endpoint saves all the information about the request into the database: remote IP address, remote port, HTTP method, protocol (HTTP/1.1 or HTTP/2), scheme (http or https), hostname, port, if the request is an upgrade, etc. It also saves all the HTTP headers.
- *Finish*: this endpoint receives the token as part of the URL and it marks as completed the test for that token. At that point new requests against the token are discarded.

Measurement Server. To collect workers' HTTP1.1 and HTTP/2 activity we decide to use H2O server as target of our tests. It is a relatively new Web server that natively supports HTTP1.1 and HTTP/2. Within its features there is the possibility to establish an HTTP/2 communication without a previous negotiation for both types of communication, secure and non-secure.

We install two servers with identical configuration, one of them is listening non secure communications and the other server is listening over the same ports but only secure communications.

Attached to them through FastCGI, a PHP application provides to the client applications all the information needed for the tests and also collects all the information produced by them. This information is stored into a MySQL database for a post-processing phase.

Measurement Client. To implement the Android client for performing HTTP/2 requests, we use and modify the library OkHttp. [5] OkHttp is an HTTP client that supports HTTP/1.1 and HTTP/2. As we explained before, we deploy two different clients: a browser client and an Android client, in this subsection of the paper we describe them and then we explain how the workers use them to perform the measurements.



HTTP/2 Tester

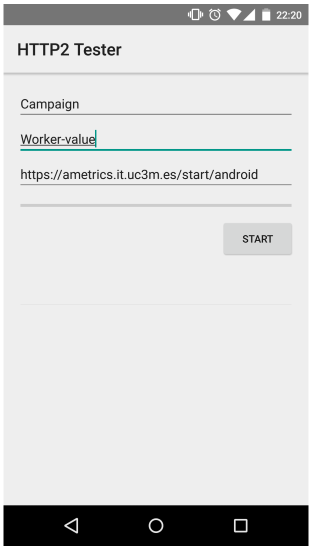
Please click start and do not close the tab until the test is completed.

https://ametrics.it.uc3m.es/start/browser/CampaignID/WorkerID

Figure 4.4: Browser App.

Browser Client. To perform the browser test, we built a Web application using PHP and JavaScript. As current browsers support HTTP/2 only over TLS, workers only performs test to secure URLs, it does not make any request using HTTP Upgrade mechanism, neither over plain text.

We start by asking the worker to connect to a webpage we provide and to start the Web application, using their own browser (i.e., Chrome, Firefox and Opera). We verify that the browser supports HTTP/2, if it is not the case we show an alert box to the worker. On start, the server returns an unique token and a list of URLs to make requests. Then, the application starts to send the h2 requests to the server and a progress bar is shown to the worker about the status of the test. To avoid overwhelming the server, we perform a request every 400ms. Once all the tests are completed, the application shows an alert box with the number of test performed and the number of successful tests (Figure 4.4).



Campaign

Worker-value

https://ametrics.it.uc3m.es/start/android

Figure 4.5: Android App.

Android Client. The Android application performs two tests, one mandatory test over mobile network and secondly, if available, another test using the WiFi connection to test fixed line. Once the test starts, the application sends extra information about the worker: local IP, network type and subtype, mobile country code, mobile network code and cell ID. The servers returns the unique token and the list of URLs to make requests on.

[5] https://github.com/square/okhttp. *OkHttp - An HTTP+SPDY client for Android and Java application*, 2015a

For each URL, the application makes first an HTTP/1.1 request, then an h2, an h2c and finally an h2c without Upgrade request over 68 different ports. A progress bar shows the status of the test to the workers. The application sends the plain HTTP Upgrade using HTTP/1.1 and it reads the response. If the response has a 200 status code, that means no Upgrade was performed and then the application continues with the next test. If the response has a 101 status code, the server accepts the HTTP/2 Upgrade[6]. The interface of the Android app is very similar to the browser implementation, basically the workers need only to press the button *Start*. Figure 4.5 shows a screenshot of the HTTP/2 Android Client's GUI.

If the request is responded by the server with a 200 status code, the application increases a counter to show the results at the end. In the case of HTTP/2 Upgrade, not only it checks for a 200 status code response, but also the application reads the JSON response looking for the protocol used and returned by the server. Once the test are completed for mobile network, the application switches on the WiFi. If the WiFi connection is available, the client performs all the tests again. Finally, the application shows an alert box with the results of the tests.

4.3.3 *Limitations*

There are a number of limitations that needs to be considered during the development of the tool.

Firstly, only last versions of Chrome, Firefox, Internet Explorer and Opera browser support HTTP/2 and only over TLS. Additionally, the Android application works in devices with the versions 4.4 and above, which it is about 58% of the market.

4.4 *The Case of the Pervasive Encryption: Results*

In this Section, we present the dataset and we evaluate the performance of HTTP and HTTPS protocols for the tested ports and results from the HTTP/2 campaign.

4.4.1 *Results from the TLS Campaign*

Dataset. In the campaigns for fixed lines, we recruit 1,165 *workers* from 53 different countries. The MAs are hosted in 286 ASes overall. 79% of the users indicated that they are connecting from *Home*, 10% from a *Company*, 6% from an *University* and 5% from a *Public Hot Spot*. Also, in the case of residential users, we collected data from *DSL* (36%), *Cable* (31%), *Fiber* (12%) and *Others* (21%). For the mobile case study, we recruit 956 *workers*, from 45 different countries and 183

[6] OkHttp does not support HTTP/2 Upgrade. To perform those tests, we use a TCP socket and implement that functionality.

ASes. 26% of the users indicated that they are using a *2G* network, the 64% a *3G* network and the remaining 10% a *4G* cellular network.

Considering that each MA performs 68 connections to our MS, we build a complex dataset for a total of 114,228 connections. When processing the data, we verify that we only observe data from unique IP addresses, to ensure that the MAs perform the test only once.

Aggregated results. We start by comparing HTTP and TLS connections that we get from the same IP address in each port for both fixed line and mobile network. We compute the percentage of errors that occur when users perform a TLS connection as

$$ERRORS(\%) = (\frac{100}{http}) \times (http - tls) \qquad (4.1)$$

where $http$ indicates the amount of HTTP connections in a specific port, while tls indicates the amount of TLS connections in the same port.

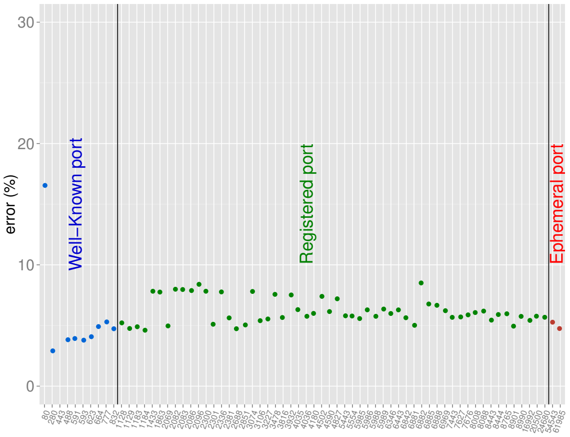
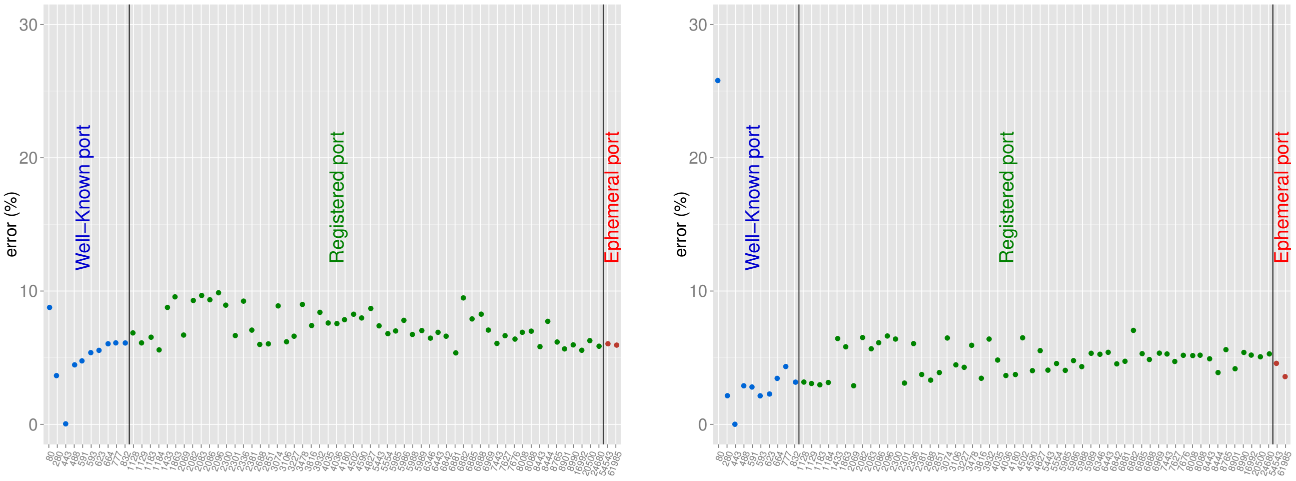


Figure 4.6: Error rate vs. port number, aggregated results.

The computed percentage of error for each port is shown in Figure 4.6. Results show that the amount of errors in port 80 is 16,5%. This is much larger that the error rate for the other tested ports, which is in average 5,8%. Next, we split the analysis for the categories of MAs we recruit, namely fixed line and mobile.

Fixed line vs. mobile network. In this Section we analyze the results from users that use a fixed line and from users connected to a cellular network to reach our server, as they declare when they complete the task, submitting the form in our web page.

Figure 4.7 (a) and Figure 4.7 (b) show the results for the campaign in which we consider users connected by fixed line compared to the percentage of errors of users that attempt to connect by a cellular



a) Fixed line

b) Mobile network

Figure 4.7: Error rate vs. port number.

network. In the case of fixed line, we observe an error rate of 6,95% in average, considering all the tested ports. Registered ports present in average an error rate slightly greater than the one we calculate for the Ephemeral and the Well-known.

To better understand the case of users behind a fixed line, we analyze specifically the category indicated by the users about the place which they are connecting. Results show that the average percentages of errors in the categories we consider for all ports are: 2% for *University*, 4,5% for *Public Hot Spot*, 5,7% for *Company* and 7,4% for *Home*. In this case, results from office or University are similar. Clearly the error rate from University is significantly lower then the other categories.

In mobile scenario the average percentage of errors is equal to the 4,54%, considering all ports. However, it is interesting to observe the behavior of middleboxes respect to TLS when port 80 is used. In this case, 25% of the users are not able to perform a TLS connection.

Figure 4.8 shows the average percentage of errors in the categories we consider for all ports. Results show a large number of errors from users connected by home. Results from office or University are similar. Clearly errors rate from University is lower then the other categories.

It is well known that cellular network operators employ a large amount of middleboxes to ensure security, traffic management, and performance optimization. Unfortunately, they are rarely transparent about middlebox policies and their impact on representative mobile workloads is poorly understood. However, we try to figure out the possible reasons of errors, in particular in port 80, analyzing the

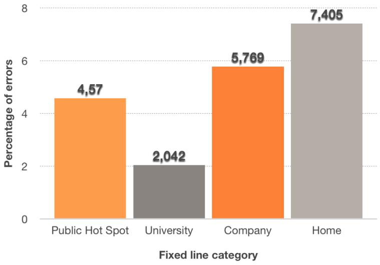
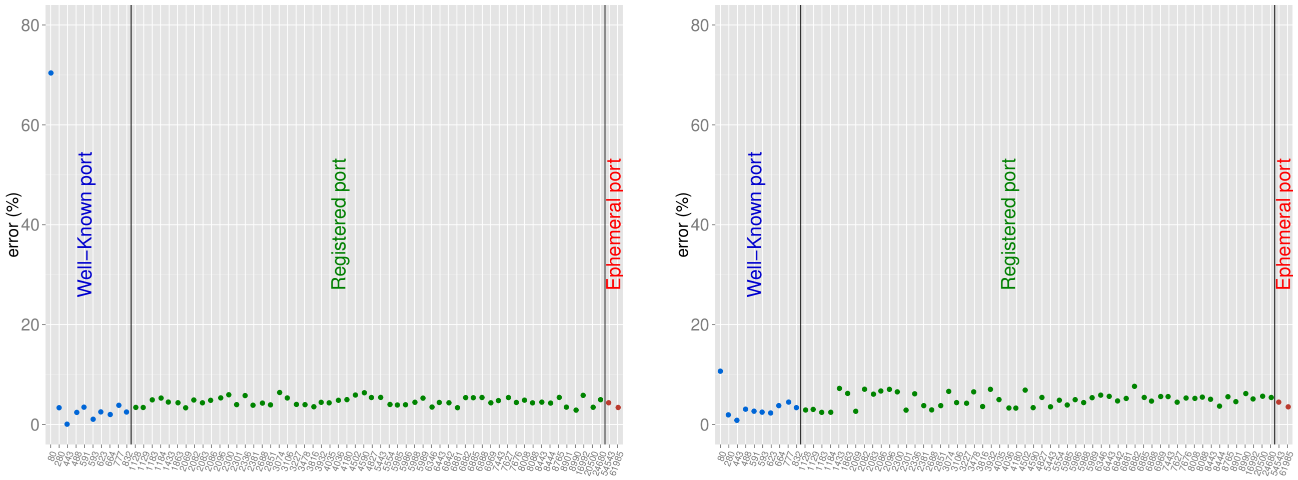


Figure 4.8: Average percentage of errors for each category in the fixed line use case for all ports.

different rate of errors between users that use proxy and users that does not use it.



a) Mobile proxy

b) Mobile non-proxy

Figure 4.9: Error rate vs. port number.

Proxies. The experiments described above highlight that, when TLS is used over port 80 in cellular network, 25% of users are not able to perform connections to our MS. In this Section, we try to understand how a proxy interacts with the communication between the MA and the MS.

Proxies are mostly used for connectivity, caching, monitoring, control and privacy. We observe two kinds of proxies in today's Internet: *transparent* and *non-transparent*. A transparent proxy, typically centralizes network traffic for security purposes before delivery to a client on the network. They transparently handle all requests to destination servers without requiring any action on the part of the client. Contrariwise, a user who is behind a non-transparent proxy knows that the proxy is being used and it can be configured. In both cases, the proxy establishes two separate connections: they terminate the TCP connection initiated by the client and they initiate a separate TCP connection between the proxy and the server. A proxy can insert into the HTTP header some standardized fields through which we are able to detect that the request has been forwarded by a proxy. It may also contain the IP address of the client. However, the administrator of a proxy server can decide whether or not to send these fields, determining the level of anonymity proxy. In our experimental setup, we can observe the HTTP headers allowing us to sometimes detect the usage of a proxy for both transparent and non-transparent category.

The HTTP proxy, depending on the anonymity that fail to provide, can be divided into: NOA (not anonymous proxy) that modifies some header sent by the browser and adds others, also it shows the real IP address of the applicant. They are very easy to recognize by the server. ANM (anonymous proxy server) proxy anonymous that does not transmit the IP address of the applicant, but modifies or adds some header. They are therefore easily recognizable. HIA (high anonymous proxy) highly anonymous proxy that does not transmit the IP address of the applicant and does not modify request headers. They are difficult to detect through normal controls. Through our methodology based on observing HTTP headers, we are able to detect if a user use NOA or ANM proxies, but we cannot detect the users that use HIA proxies.

In the case of fixed line we find that the 1% of the users use a proxy. In the case of mobile the percentage increases to 25%.

In Figure 4.9 we compare the rate of successful TLS connections for users we detect using a proxy (a) and for users that do not (b) in mobile network scenario. We observe that in the case proxies are present, the error rate of TLS in port 80 is 70% and the error rate in all other ports is 4,23% in average. When proxies are not used the behavior of the connections is similar to the case of the fixed line.

It is interesting to note the different rate of errors respect to the different networks through the users are connected, when a proxy is used. Results demonstrate that when users use a proxy the average percentage of errors in 2G is 49%, in 3G is 80% and in 4G network is 100%.

Packets analysis. To better understand how proxies or other middleboxes behavior impacts the performance of the TLS protocol in unconventional ports, we focus on the packet analysis, splitting the analysis for fixed line, mobile and for users that use or not a proxy.

We observe that in a large number of cases, for the TLS connection, the server does not even receive the TCP SYN packet from the MA.

When we analyze the packet-level traffic our analysis tests for the following possible middleboxes behavior:

- Connection is reset by the middlebox: we do not receive the SYN packet;

- Connection is reset before the TCP handshake: we receive the SYN packet.

Table 4.2 refers to the percentage of SYN we receive when users try to establish a TLS connection to our MS from fixed line use case and from mobile network, considering all port and particularizing the analysis for port 80 (labels *All*, *Port 80*). Moreover, in the case of

Analysis	Fixed Line		Mobile	
	SYN(%)	NO SYN(%)	SYN(%)	NO SYN(%)
All	96,8	3,2	36	64
Port 80	88,3	11,7	27,7	72,3
Proxy			22,2	77,8
Non-proxy			12,7	87,3
Proxy (80)			9,6	90,4
Non-proxy (80)			36,4	63,6

Table 4.1: Packets analysis

mobile network we particularize the analysis for proxy/non-proxy case (labels *Proxy*, *Non-proxy*, *Proxy (80)*, *Non-proxy (80)*).

We observe that in the case of proxies, 90% of the SYN packets are missing. While this may seem non causal at first (as the SYN packet is forwarded before the middle box actually knows whether this is a regular HTTP connection or a TLS connection), proxies usually wait until they receive the GET from the client to establish the connection to the server in order to apply their policies. This explains why in the case of TLS, we miss a high number of SYN packets.

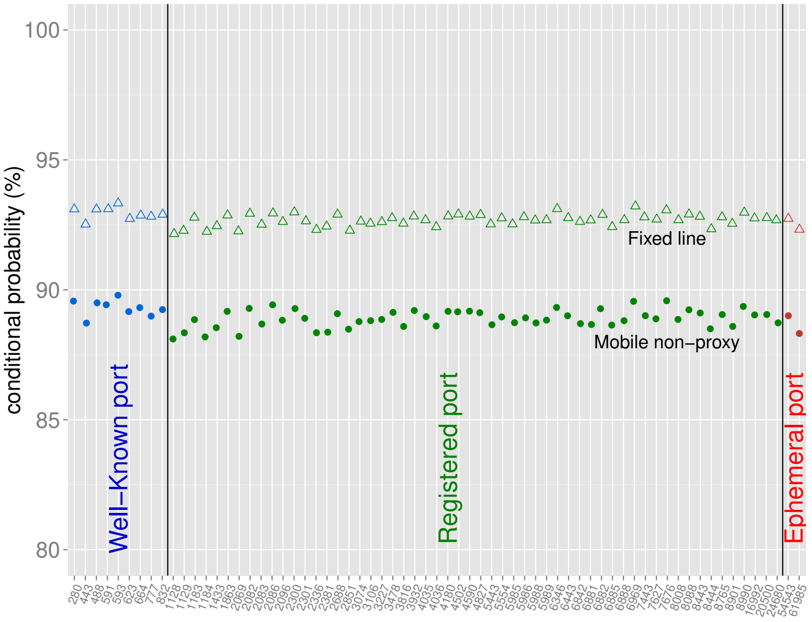


Figure 4.10: Conditional probability vs. port number.

Consistent filtering. In this Section, we try to understand if the filtering of TLS is consistent across the different ports for a given MA. In other words, if the TLS connection fails in a given port, how likely is that it will fail in other ports. In order to quantify this, we estimate the conditional probability of failure in a given port X given that the

TLS connection in port 80 has failed. We choose the port 80 as it is in general a port with high failure rate. We estimate the aforementioned conditional probability for the case of fixed line and for the case of mobile network without proxies (Figure 4.10). The case of mobile network with proxy is not very interesting, as there is a much larger failure rate in the port 80, so the conditional probability will look like the error rate in the different ports.

Figure 4.10 shows the percentage of errors in other ports, when an error occurs in port 80, considering the fixed line case and the mobile use case when users do not use proxies. We can see that the estimated conditional probability is around 90% in both cases (slightly higher in the fixed line case), implying that when the TLS connection fails in port 80, it is very likely that it will fail in the other ports. It is reasonable then to guess that the same behavior will occur in the other ports beyond the ones we have measured.

4.4.2 HTTP/2: To Encrypt or Not to Encrypt?

In this subsection, we first present the data-set collected; next, we analyze the interaction between HTTP/2 and the Internet ecosystem, differentiating between fixed line and mobile network. Our main goal is to quantify how likely HTTP/2 connections are to be (un)successful. We thus introduce the following metric:

$$ERRORS(\%) = (\frac{100}{http}) \times (http - http2) \tag{4.2}$$

where “http” and “http2” indicate the number of successful HTTP/1 and HTTP/2 connections on a given port. We leverage HTTP/1 connections to eliminate non-HTTP/2 related phenomenons, such as connectivity issues.

We perform 4 different tests: using H2, H2 without ALPN negotiation, H2C and H2C without Upgrade. In this Section we present the results for the browser-based campaign over fixed line, the application-based campaign over fixed line (WiFi) and the application-based campaign over mobile network.

Data Set. We run the browser-based campaign for 4 days and we pay each worker 0.2$. Instead, we run the application-based campaign for seven consecutive days, and we pay each worker 0.4$, on average, as the cost of a campaign depends on the worker’s country. Overall, we recruit 628 workers: 306 workers participate to the browser-based campaign, while 322 workers participate to the application-based one. Out of these 322 workers, only 49 workers had both mobile and WiFi connectivity available, and thus generate fixed line results (over WiFi).

In the browser-based campaign, each worker attempts two connections (HTTP/1 and H2) for each of the 67 available ports at our servers. In the application-based campaign, each worker attempts four connections per port (HTTP/1, H2 without ALPN, H2C and H2C without Upgrade). It follows that our data-set consists of 54,136 unique connections from fixed line, and 86,296 connections from a mobile network.

Mobile users are partitioned among 2G (17%), 3G (71.8%), and 4G (3,6%); we could not detect the access network used by the remainder 7,6% of the mobile workers. Overall, workers are distributed among 38 countries (cf. Figure 7.3).

Fixed line. In this subsection, we present the results for the fixed line case study. First, we show the percentage of workers that are not able to perform an H2 connection using their own browser; next we show the percentage of errors when workers use the Android application over WiFi connection.

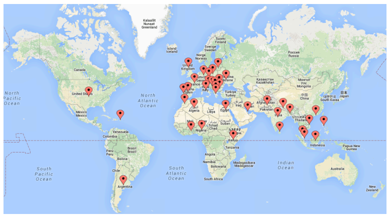


Figure 4.11: Distributed vantage points map.

Browser-based Campaign. Figure 4.12a shows the error rate as a function of the port number; these results were collected via a regular browser (running our Web application) using a fixed access. We asked explicitly the workers to perform the test using their personal computer and fixed line. On average, we measure an error rate of only 2%; such low number is to be expected, since these results are originated by regular (encrypted) H2. However, we also measure high error rates for the following ports: 80 (4.90%), 593 (6.90%) and 5554 (5.55%). These are well-known ports used for, respectively, the Web (80), remote procedure call over HTTP (593) and SGI ESP HTTP (5554). It follows that, likely, middleboxes are instrumented to monitor traffic on these given ports, and somehow manage to interact even with encrypted H2 traffic. Strengthening such thesis, we also observe an error rate of 0% on port 443, the default port used by encrypted HTTP/1 (HTTPS) which middlebox are likely ignoring.

To further understand this result, we resort to pcap traces collected at our servers. Table 4.2 categorizes the errors detected for workers that participated to the browser-based campaign. When an error is detected on port 80, more than 80% of the time even the TCP SYN packet is not received. In this case, an anonymous HTTP proxy is likely used by the ISP. HTTP proxies, in fact do not forward the SYN packet before to know whether this is a regular HTTP connection and they usually are employed to inspect traffic over port 80.

Considering port 593 and port 5554, SYN is not received for more than 85% of the time. Since we do not detect this kind of error in the case of Android applications (cf. Subsection 4.4.2) and in the knowledge that these ports are usually filtered due to Windows Virus and worms [7], we can conclude that in the case of port 80 errors are due to middleboxes interaction, while in the case of port 593 and port 5554 the errors are due to the operating system. Inspecting the user agent field for that workers in fact, it is confirmed that they are using Windows operating system. We also note a case in which a middlebox do not recognize the ALPN extension over port 80, and we receive a normal HTTPS connection (labeled *No ALPN*). There are some random ports for which the server sends a reset (RST) flag after the *client key exchange* procedure and the workers are not able to send the data (labeled *No data*).

Table 4.2: Browser-based campaign errors classification

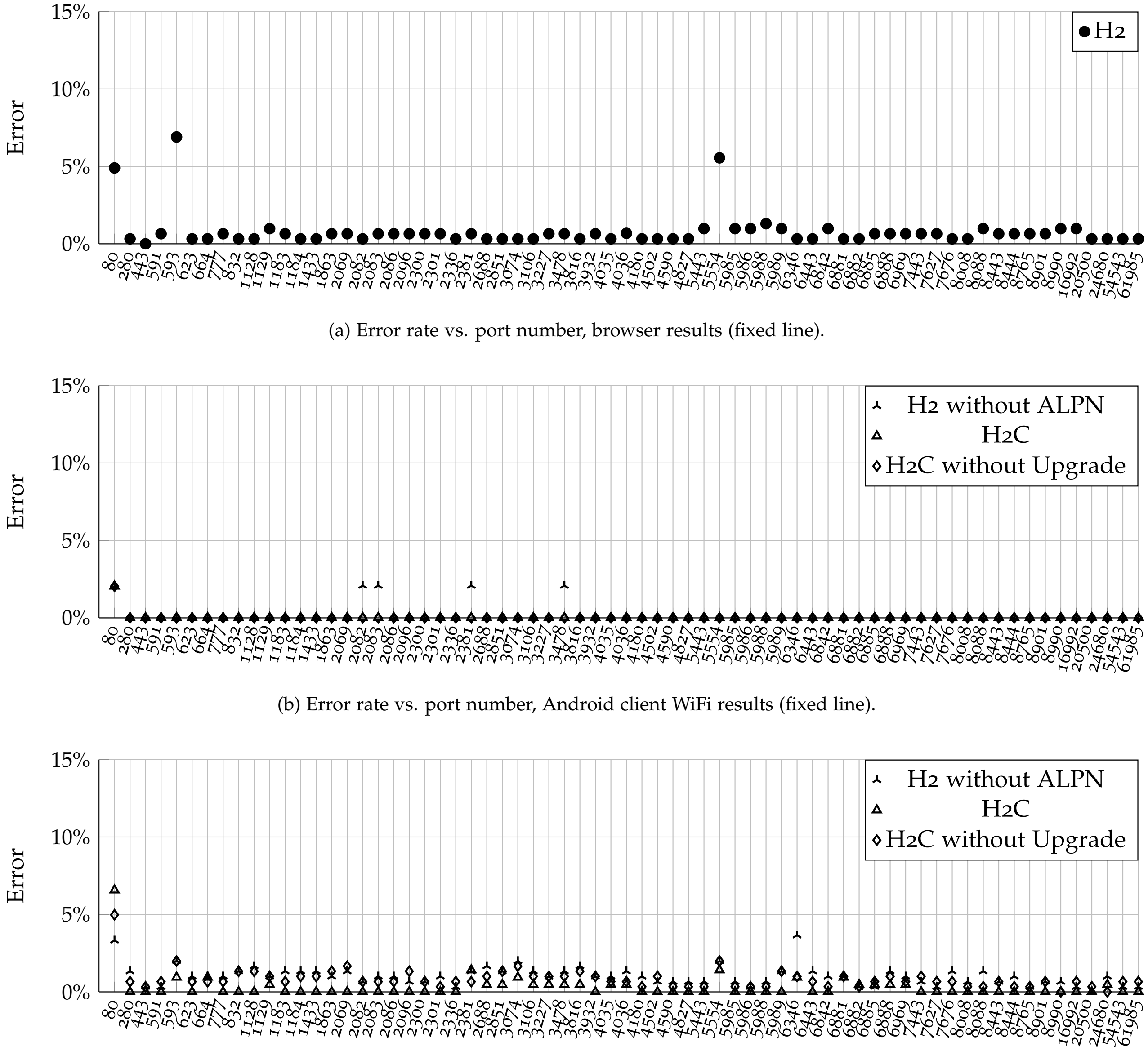
H2 behavior	Number of workers (%)
No SYN 80	81.81
No SYN 593	86.66
No SYN 5554	91.66
No ALPN	9.09
No data	100

Application-based Campaign. Figure 4.12b shows the error rate as a function of the port number, differentiating between H2 without ALPN negotiation, H2C, and H2C without Upgrade. These results were generated via our Android application, while using WiFi connectivity. Overall, over port 80 we detect an error rate of 2.04% for both H2C and H2 test cases. As the browser-based campaign this port is affected by middleboxes that expect HTTP protocol. For that particular port in fact, we do not receive the SYN packet. Some errors appear on few random ports (port number 2082, 2083, 2381 and 3478) and only for H2 test case. In order to better understand the communication issue for those ports we leverage the pcap traces. We detect connectivity problems during the TLS handshake. As in the browser-based campaign case the server resets the communication after the *client key exchange*, which excludes a middlebox interference.

Mobile Network. Figure 4.12c shows the error rate as a function of the port number, differentiating between H2 without ALPN negotiation, H2C, and H2C without connection Upgrade. These results were generated via our Android application, while using mobile connectivity.

As above, the overall error rate is quite low: 0.58%, on average. Differently from above, we observe a noticeable trend: 7% of the H2C connection attempts fail over port 80! This result is remarkable, since this is the most common usage of H2C we would expect. The error rate reduces to 4.3% when we assume direct H2C, *i.e.*, where a con-

[7] http://www.speedguide.net/port.php?port=593. *Port 593 filter*, 2015



(a) Error rate vs. port number, browser results (fixed line).

(b) Error rate vs. port number, Android client WiFi results (fixed line).

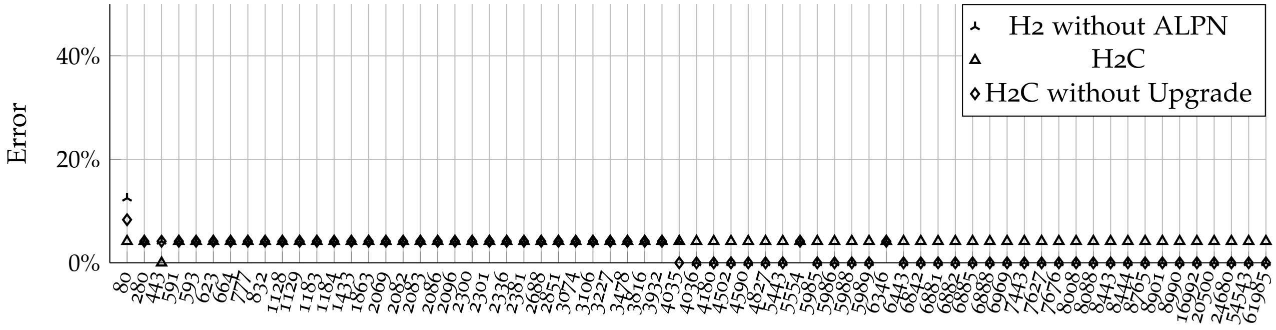
(c) Error rate vs. port number, Android client results (mobile network).

Figure 4.12: Error rate vs. port number.

nection Upgrade was not performed. It follows that *H2C might face an additional challenge in its adoption, due to the presence of middleboxes that, ironically, very much like encrypted content.*

In order to further understand the latter result, we investigate the pcap traces collected at our servers. In the specific case of port 80, we observe that about 80% of the SYN packets are not received by our servers. This is due to a more common usage of middleboxes in mobile networks compared than in fixed lines. For example, proxies wait until reception of the first GET request from a client before opening a connection with the server; they can therefore discard a protocol that is not the expected on that port.

We also verify if the filtering of HTTP/2 is consistent across the different variation of the protocol for a given worker over port 80. Basically, we compute the probability of failure for a given variation of HTTP/2 (e.g., H2C) respect to other variations (e.g., H2C without Upgrade and H2). Specifically, if a worker fails an H2C connection, how likely is that it will fail an H2C without Upgrade and an H2 connection. Results show that when a worker fails an H2C connection, he fails also the H2C without Upgrade and the H2 connection.



(a) Error rate vs. port number, NATs Android client results (mobile network).

Figure 4.13: Error rate vs. port number.

Proxies. In order to better understand the results in mobile environment, we extend our study by analyzing proxy behavior over port 80. We consider the HTTP header of the workers that produced errors for HTTP/2 without ALPN, H2C and H2C without Upgrade and we try to identify the use of proxies along the path for those test cases.

Today's proxies can be divided into two groups: anonymous and non-anonymous. Anonymous proxies are not detectable by servers, non-anonymous proxies leave traces of their presence in the HTTP headers.

To discover if workers are using a proxy, we check for the existence of the following headers in the HTTP requests:

- `CLIENT-IP`
- `FORWARDED`
- `FORWARDED-FOR`
- `FORWARDED-FOR-IP`
- `PROXY-CONNECTION`
- `VIA`
- `X-FORWARDED`
- `X-FORWARDED-FOR`
- `X-GATEWAY`
- `X-NETWORK-INFO`

Protocol	# of errors (%)	# of proxies(%)
H2 without ALPN	3.32	18
H2C	6.6	24
H2C without Upgrade	5	12.5

Table 4.3: Android-based mobile campaign proxy errors.

Table 4.3 refers to the percentage of workers that produce an error and are behind a proxy, for H2 without ALPN, H2C and H2C without Upgrade in mobile network case studies. Considering the case of H2C results show that 24% of errors are due to proxies. Some of these requests (40%) do not arrive to our server and the 60% of them do not Upgrade to HTTP/2. When we try to send the H2C request without the Upgrade the percentage of errors due to a proxy is 12.5%. Consequently, a direct H2C request (without Upgrade) is more successful that an H2C connection. In this case, in fact, proxies does not recognize the Upgrade and changes the HTTP/1 101 Switching Protocols header to a normal HTTP/1 request.

4.5 *Privacy Consideration*

To preserve the privacy of the workers we recruited through the Microworkers crowdsourcing campaigns, we give sufficient information in order that a worker would be able to make a choice of whether or not to participate to the test. This information has been written in a way that was understandable to the people who have been approached as participants. Even though the goal of this paper is understand the ISPs infrastructure and does not focus on human personal information, the data collected on our server are accurately protected and shared once they have been rendered anonymous, for example sensitive information have been replaced with indirect identifiers (e.g. numbers). Accepting to run the test workers give the informed consent to share such data to the research community. Once the campaign is run, we provide to the worker the clarification of the main objective of our work and some references to our research. We also publish the source code we use to perform the tests to be checked by the workers that run it in their own machine.

5 TCP Fast Open: Initial Measurements

Given the transition of mobile technologies from GSM to LTE-advanced, users rely more and more on mobile broadband networks. Mobile devices are becoming pervasive, raising the expectation of users for faster connectivity in a mobile scenario. In this context, employing optimizations solely focused on eliminating round-trip protocol messaging is central to surmounting critical performance issues. TFO is one such solution, focused on making TCP faster by eliminating one RTT in the three-way hand shake (3WHS).

TFO [1] allows to send data directly in the SYN packet. Latency sensitive applications such as gaming, audio/video calls, financial transaction and embedded advertisement can benefit from it. The 3WHS employs one RTT before the server's application receives the data and two RTTs for the server response. TFO should speed up the exchange of information between client and server and reduce of one RTT the 3WHS. Sending the initial SYN with data, TFO could reduce the loading of pages up to 40%.[2]

Unfortunately, the existence of middleboxes can influence the normal operation of TFO. One major criticism of middleboxes is that they might filter traffic that does not conform to expected behavior. For example, they may drop SYN packets with data or with unknown TCP options. In this work we force TFO connections from different vantage points, in different ports to evaluate the performance impact of these drop behaviors.

5.1 TFO Operation

In this subsection we describe the TFO standard operation. Figure 5.1 and Figure 5.2 show the operation of TFO when the client connects to the server for the first time. At the begging of a TCP connection, the client creates a SYN packet with the TFO option enabled. IANA assigned to TFO the option number 34, however, Linux currently uses the old experimental TCP option 254. This option corresponds to a TFO cookie request. Because no TFO connections have been

[1] S. Radhakrishnan, Y. Cheng, J. Chu, A. Jain, and B. Raghavan. Tcp fast open. In *Proceedings of the Seventh COnference on Emerging Networking EXperiments and Technologies*, CoNEXT '11, pages 21:1–21:12, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-1041-3

[2] S. Radhakrishnan, Y. Cheng, J. Chu, A. Jain, and B. Raghavan. Tcp fast open. In *Proceedings of the Seventh COnference on Emerging Networking EXperiments and Technologies*, CoNEXT '11, pages 21:1–21:12, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-1041-3

established, the cookie is empty. Through this procedure the client indicates that it supports TFO and it would like to receive a cookie. In this first connection, no data are sent into the SYN packet. Practically, connections benefit from TFO only after an initial TCP connection is established (Figure 5.1).

Once the server receives the SYN with the TFO cookie request and it supports TFO, it generates a message authentication code (MAC) for the client, using the IP address of the client and its secret key. if it does not support TFO it will ignore the TFO option and establish a standard TCP connection. Then, it sends the cookie to the client in the SYN-ACK packet, using the same TFO option and it establishes a standard TCP connection (Figure 5.1).

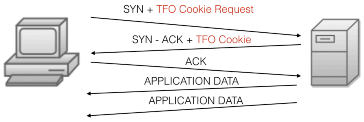


Figure 5.1: TFO cookie request.

The client, receiving the TFO cookie, caches it and the correspondence to the server's IP that sent it. The client will use the cookie to establish a TFO connection to the same server in the next connection. When the client tries to connect to the server again, it sends a SYN packet containing the TFO option enabled, the cookie previous cached that corresponds at the server's IP, and the application data. When the server receives the SYN packet with the TFO option, the cookie and the data, it generates the cookie and compares it to the cookie sent by the client. If the cookie matches, the server sends a SYN-ACK, acknowledging the SYN and the data, consecutively it sends the data to the client. If the cookies do not match, the server only acknowledges the SYN, including the new cookie to use next time. Once the client receives the SYN-ACK, it checks the packet to ensure that the server acknowledges also the data. If it is not the case, the client sends the ACK to the server and successively it sends a packet containing the application data, switching in a standard TCP connection (Figure 5.2).

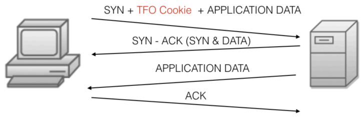


Figure 5.2: TFO cookie request.

5.2 *TCP Fast Open: Experimental Setup*

In this section we describe the methodology to measure in the wild whether TFO is supported by the Internet paths. To this end we recruit TFO users from Microworkers crowdsourcing platform. We create a tool called ExploreTFO that allows users to connect to our TFO server.

TFO requires client and server kernel support. To this purpose we create a tool namely TFOExplorer.

TFOExplorer is composed by a client (from now on, TFO Client (TC)), a target server that supports TFO protocol (from now on, TFO Server (TS)) and a script needed to be run in the TC machine (from now on, TFO Script (TSC)).

In order to test the successful of the TFO connections through the Internet we recruit TCs from the Microworkers crowdsourcing platform. We collect and analyze the results from 46 TCs that try to establish TFO connections in a large number of ports to our TFO Server (TS) and over port 80 and 443 to Google servers.

Our methodology allows us to control the TCs and the TS at the same time. Therefore, in the eventuality of a connection failure we are able to know if the TC attempted to connect to our server.

TFO Server (TS). Server side, TFO protocol is available in Linux 3.7+. Moreover, nginx 1.5.8+ web server already supports it. We configure a dedicated nginx web server version 1.6.2 with TFO supporting in 68 different ports, namely 10 well-known ports, 56 registered ports and 2 ephemeral ports. We expect middleboxes along the path change their behavior depending on the unusual application port, for this reason we consider ports normally used by HTTP applications. Moreover we store and analyze in detail the server side packet exchanges. If an error occurs in some ports, we analyze the packets exchanges to monitor the behavior of the TCP with respect to the middleboxes active along the path.

TFO Client (TC). Client side, TFO protocol is available in Linux 3.7+ or Android 5.0 (Lollipop). Browsers supporting TFO are Chrome/Chromium on Linux, ChromeOS, and it needs to be enabled manually. Through an appropriate script we ensure that TCs have a version of Linux > 3.7 to support TFO.

TFO Script (TSC). The TSC includes a TFO client, written in C programming that allows workers to establish TFO connections to the servers that support it and to send a HTTP GET directly into the SYN packet.

The TFOExplorer allows us to control the client, capturing automatically the packets exchange in the TC's machine and sending the files containing the results of the experiment directly to our server to be analyzed.

The common procedure is as follows. We ask the workers to connect to our server through a normal browser over port 80 and to follow the instructions in our web page. Basically, they need to download the TSC and run it in their Linux machine, using root privileges. Once completed, a code appears in their screen, this will be the proof for Microworkers payment.

5.3 *TFO: Initial Results*

We collect and analyze the results from 46 users from 18 different countries and 22 different ISPs. Each user tries to establish TFO connections in a large number of ports to our TS. Considering that each TC performs 68 connections to our TS three times (twice to perform a complete TFO connection and one to send data in the initial SYN packet), we build a dataset for a total of 9,568 connections 1. Table5.1 shows the results from workers that attempts to connect to our TS, using TFO. Analyzing the data, we check for the following behaviors: TC is able to perform a TFO connection (label Successful); middleboxes drop packets with unknown TCP options and we receive the SYN without option (label No option SYN); middleboxes drop packets with unknown TCP options and we do not receive the SYN without option (label No option no SYN); middleboxes drop packets with data in the SYN packet and we receive the SYN without data (label No data SYN); middleboxes drop packets with data in the SYN packet and we do not receive the SYN without data (label No data no SYN).

TFO behavior	Number of workers	Number of workers (%)
Successful	19	41,3
No option SYN	18	39,13
No option no SYN	0	0
No data SYN	8	17,39
No data no SYN	1	2,18

Table 5.1: TFO connection

TFO behavior	Number of workers	Number of workers (%)
Successful	23	50
No data SYN	23	50
No data no SYN	0	0

Table 5.2: Data in the SYN

Results show that only the 41,3% of the packets with the TFO experimental option are able to arrive to our TS. The 39,13TFO allows for payloads to be carried in SYN frame. Once the client has the cookie, it tries to send the data in the SYN. In this case, only the 67,86% of the packets are received by our TS. The remainder of the packets arrive with no data in the SYN. In this case, middleboxes block the connection completely after processing the SYN packet with data, causing a connection timeout and the TC switches to a standard TCP connection, retransmitting the SYN packet with no option and no data. If, for example TFO connection fails in port 80, then it fails in all other ports. We detect only three TCs that succeed to perform a TFO connection in the Well-known and the Ephemeral ports, but that fail in some Registered ports. Table5.2 shows the results considering that case when TCs send data in the initial SYN with no TFO option set. In this case, the 50% of the SYN packets containing data is received by our TS, the rest of the packets are dropped by middleboxes, forcing a SYN retransmission client side.

6 ECN++

Expicit Congestion Notification (ECN) [1] is a way to mark packets to indicate that the capacity of a link is approaching exhaustion. ECN was standardized as a straight replacement for loss signals, but increasingly ECN is also being recognized as critical for low delay. [2]

Despite early evidence of its positive impact Salim and Ahmed [2000], a succession of unfortunate incidents stalled ECN deployment for 15 years: some firewalls treated all TCP/ECN connection attempts as port scanning attacks Floyd [2002]; TCP/ECN connection attempts were mistakenly discarded by certain home router models and one popular model crashed Thaler et al. [2007]; and when routinely wiping the Diffserv field between networks, a bug wiped the IP/ECN field too.

In recent years, ECN adoption on end-systems has accelerated, with the majority of servers supporting ECN Trammell et al. [2015]. Currently, motivated by an increasing need to reduce queuing delay in modern networks, solutions such as DCTCP Alizadeh et al. [2010] in data centers or L4S [3] in public networks are further driving interest in ECN. In 2016, Apple enabled client negotiation of TCP ECN in a random subset of iOS and macOS devices. In March 2017 they presented a measurement study Bhooma [2017] showing almost universal path traversal support for ECN. Other recent measurement studies [4], [5] are consistent with these findings.

If both sender and receiver support ECN, they set up a feedback loop between them with whatever transport layer protocol they are using (TCP, RTP, etc) . Then at the IP layer the sender sets packets as ECN-capable, which indicates that the end-systems would prefer the network to mark rather than drop packets to indicate approaching congestion.

This work focuses on using TCP over ECN in IP. The original ECN specification for TCP K. Ramakrishnan [2001] prohibited retransmitted packets and control packets from using ECN. This was unfortunate, because TCP performance, particularly short-flow completion

[1] D. B. K. Ramakrishnan, S. Floyd. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3360 (Best Current Practice), Sept. 2001. URL https://tools.ietf.org/html/rfc3168. Accessed on 2019-02-02

[2] M. Welzl and G. Fairhurst. The Benefits of using Explicit Congestion Notification (ECN). Internet Draft draft-ietf-aqm-ecn-benefits-08, Internet Engineering Task Force, Nov. 2015. URL https://tools.ietf.org/html/draft-ietf-ecn-benefits. (Work in Progress)

[3] B. Briscoe (Ed.), K. De Schepper, and M. Bagnulo. Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture. Internet Draft draft-ietf-tsvwg-l4s-arch-00, Internet Engineering Task Force, Apr. 2017. URL https://tools.ietf.org/html/draft-briscoe-tsvwg-l4s-arch. (Work in Progress)

[4] S. Bauer, R. Beverly, and A. Berger. Measuring the State of ECN Readiness in Servers, Clients, and Routers. In *Proc ACM SIGCOMM Internet Measurement Conference (IMC'11))* ,, pages 171–180, 2011

[5] B. Trammell, M. Ḱuhlewind, D. Boppart, I. Learmonth, G. Fairhurst, and R. Scheffenegger. Enabling Internet-Wide Deployment of Explicit Congestion Notification. In *In Proc Passive & Active Measurement (PAM'15) Conference*, 2015

time, is much more sensitive to loss of certain control packets, such as the SYN at the start of a connection.

A new proposal called ECN++ [6] proposes safe ways to remove all the original prohibitions on using ECN on each type of TCP packet. As with any new protocol, ECN++ can experience deployment problems, either because existing networks and servers protect themselves against out-of-the-ordinary behavior, or because optimizations have been built around a narrow and unchanging interpretation of the way protocols work [7]

In this work we exploit the great potential of the MONROE measurement platform for informing on the good functionality of ECN++ on mobile networks, which are plenty of middleboxes, and its interaction with the top 100k Alexa servers.

We do not just check for correct transitions of protocol fields, but also characterize the TCP behavior of the servers we test. We leverage the TBIT methodology, introduced by Padhye et al. [8] TBIT is a tool designed specifically to characterize the TCP behavior of web servers – to answer questions such as the distribution of TCP's initial value of the congestion window on servers in the wild. Spoofing congestion markings has also been proposed for receiver compliance testing by operational servers. [9]

6.1 ECN Background

Explicit Congestion Notification (ECN) is a standardized mechanism [10] to signal congestion by marking a field in the Internet Protocol headers. ECN [11] can be deployed in any buffer in any network element. A bottleneck buffer will mark the standardized ECN field [12] increasingly often as it detects early signs of increasing load. These ECN markings then propagate to all the hosts receiving data through the buffer. In turn, the receivers feed back these signals to the respective hosts that are sending them data. The aim is that data senders will use this feedback to regulate the load on the bottleneck buffer.

ECN is intimately tied to Active Queue Management (AQM) technology. With AQM, buffers drop a small proportion of packets at the first sign of queue growth to fool TCP senders into thinking the buffer is full and backing off. ECN allows AQM to keep the queue short without dropping packets.

6.1.1 ECN in IP

The ECN field has two bits, hence it supports the 4 codepoints in Table 6.1. Before using ECN, the standard requires a data sender to have checked that the receiver has logic for feeding back ECN

[6] M. Bagnulo and B. Briscoe. ECN++: Adding Explicit Congestion Notification (ECN) to TCP Control Packets. Internet Draft draft-bagnulo-tcpm-generalized-ecn-04, Internet Engineering Task Force, May 2017. URL https://tools.ietf.org/html/draft-bagnulo-tcpm-generalized-ecn. (Work in Progress)

[7] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda. Is it still possible to extend TCP? In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 181–194. ACM, 2011

[8] J. Padhye and S. Floyd. On Inferring TCP Behavior. *Proc. ACM SIGCOMM'01, Computer Communication Review*, 31(4):287–298, Oct. 2001. (Aka. Identifying the TCP Behavior of Web Servers)

[9] R. Sherwood, B. Bhattacharjee, and R. Braud. Misbehaving TCP Receivers Can Cause Internet-Wide Congestion Collapse. Tech report UMD-CS-TR-4737, UMD, 2005

[10] D. B. K. Ramakrishnan, S. Floyd. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3360 (Best Current Practice), Sept. 2001. URL https://tools.ietf.org/html/rfc3168. Accessed on 2019-02-02

[11] S. Floyd. TCP and Explicit Congestion Notification. *ACM SIGCOMM Computer Communication Review*, 24(5):10–23, Oct. 1994. (This issue of CCR incorrectly has '1995' on the cover)

[12] D. B. K. Ramakrishnan, S. Floyd. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3360 (Best Current Practice), Sept. 2001. URL https://tools.ietf.org/html/rfc3168. Accessed on 2019-02-02

Codepoint	IP/ECN field	Meaning
not-ECT	00	Not ECN-capable transport
ECT(0)	10	ECN-capable transport
ECT(1)	01	
CE	11	Congestion experienced

Table 6.1: The ECN field in IP.

markings. Each transport protocol (TCP, SCTP, QUIC, etc.) does this differently. If at least one end does not support ECN, there is not an 'ECN-capable transport' (ECT) so the sender must set not-ECT in all packets, which is also what legacy hosts send by default. According to [13], ECT(0) and ECT(1) are equivalent and either can be set by the sender to signify the packet is ECT, which means "ECN-capable but not ECN-marked".

Network elements have to check the ECN field before using ECN. If it is not-ECT (00), they must use drop rather than the CE codepoint to indicate congestion. If the ECN field is non-zero, any buffer can set the CE codepoint (11) increasingly often to indicate increasing congestion.

Feedback mode	Packet	TCP ECN flags		Allowed IP ECN	
		Description	flags	ECN	ECN++
Non-ECN	All	ECN disabled	000	00	00
ECN	SYN	ECN setup	011	00	00
	SYN/ACK	ECN setup	001	00	XX
	Data	Regular	000	XX	XX
		Echo CE	001	XX	XX
		CWnd Reduced	010	XX	XX
	Control	Same as data packet		00	XX
AccECN	SYN	AccECN setup	111	00	XX
	SYN/ACK	AccECN setup	010	00	XX
		with CE Echo	110	00	XX
	Data	CE counter	XXX	XX	XX
	Control			00	XX

Table 6.2: Allowed IP ECN field for all types of TCP packet in all three feedback modes. X means 0 or 1.

6.1.2 *ECN in TCP*

Three flags in the main TCP header are assigned to ECN. They are called NS, CWR and ECE for Nonce Sum, Congestion Window Reduced and Echo Congestion Experienced. These names are not descriptive of their usage in most circumstances, so this paper will represent them as a 3-bit codepoint, as shown in the 'flags' column of Table 6.2.

Table 6.2 succinctly supports the following explanations of standard ECN (this Section), Accurate ECN (Section6.1.3) and ECN++

[13] D. B. K. Ramakrishnan, S. Floyd. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3360 (Best Current Practice), Sept. 2001. URL https://tools.ietf.org/html/rfc3168. Accessed on 2019-02-02

(Section6.1.4). For the present explanation of standard ECN, ignore the AccECN rows at the bottom and the ECN++ column on the right. Also, the NS TCP flag is not used for standard ECN (always zero).

For the example of an ECN client contacting an ECN server, the table should be read down the rows as follows. The client sends an 'ECN setup SYN' with TCP ECN flags 011 and the server responds with an 'ECN setup SYN/ACK' with TCP ECN flags 001. Only 00 is allowed in the IP ECN field of both these handshake packets, as shown in the 'Allowed IP ECN' column.

Once the TCP connection is established, ECN feedback proceeds independently in either direction for the two half-connections. Regular data packets have 000 in the TCP ECN flags and, according to the IP ECN column, the data sender can set either ECT codepoint (XX means theoretically any of the 4 values in Table 6.1 are allowed). Then, if congestion is experienced along the path, the buffer will set the CE codepoint on some of these packets.

The data receiver will then feed this back by setting the Echo CE (ECE) flag in the TCP header of acknowledgement (ACK). The data sender then confirms receipt of a new ECE flag by setting the CWR flag on the next segment, which confirms (or at least claims) that it has reduced its congestion window (cwnd). For reliability against loss of an ECE message, the data receiver is required to set the ECE flag repeatedly on every ACK until it receives the CWR, one round trip later.

6.1.3 Accurate ECN

Because the original ECN scheme repeats the ECE flag for a whole round trip for reliability, more than one CE mark within a round trip cannot induce any more feedback. Since 2010, it has become well-known [14], [15] that queuing delay can be reduced to extremely low levels if more accurate feedback gives the extent, not just the existence, of CE-marking. This was first proved in Data Centres with DCTCP Alizadeh et al. [2010], and subsequently for the public Internet using the new Low Latency Low Loss Scalable throughput (L4S) technology. [16]

AccECN Briscoe et al. [2016] adds more accurate ECN feedback to TCP. We introduce it in our testing, but we do not expect to find it in the wild yet because standardization is not yet complete.

The handshake for a client and server supporting AccECN feedback can be seen in the AccECN setup SYN and SYN/ACK rows of Table 6.2. We will not step through the table again, except to highlight differences. The 110 combination of TCP ECN flags allows feedback on the SYN/ACK in the event that the SYN arrives at the

[14] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data Center TCP (DCTCP). *Proc. ACM SIGCOMM'10, Computer Communication Review*, 40(4):63–74, Oct. 2010

[15] B. Briscoe (Ed.), K. De Schepper, and M. Bagnulo. Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture. Internet Draft draft-ietf-tsvwg-l4s-arch-00, Internet Engineering Task Force, Apr. 2017. URL `https://tools.ietf.org/html/draft-briscoe-tsvwg-l4s-arch`. (Work in Progress)

[16] B. Briscoe (Ed.), K. De Schepper, and M. Bagnulo. Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture. Internet Draft draft-ietf-tsvwg-l4s-arch-00, Internet Engineering Task Force, Apr. 2017. URL `https://tools.ietf.org/html/draft-briscoe-tsvwg-l4s-arch`. (Work in Progress)

server with a CE mark, otherwise the SYN/ACK uses 010. This enables the SYN to be ECN-capable (see ECN++ in Section6.1.4). Once an AccECN connection is established, an AccECN data receiver uses the ECN TCP flags as a 3-bit counter to continually repeat feedback of a count of how many CE-marked packets it has received over the half-connection. AccECN also uses a TCP Option for feedback in bytes, but the 3-bit feedback is sufficient if the option does not survive traversal over the network.

Note that if a server supports a more recent mode of ECN feedback than that requested in the client's SYN, the server downgrades its response to match the client. For instance, an AccECN server will respond to an ECN client as if it is an ECN server. Or, it will respond to a non-ECN client as if it is a non-ECN server. Similarly, if the client supports a more recent mode, it will recognize the server's legacy SYN/ACK response and downgrade itself.

6.1.4 ECN++

When ECN was first standardized, SYNs and SYN/ACKs were precluded from being ECN-capable at the IP layer. IP/ECN was similarly prohibited for pure ACKs, window probes, and retransmissions (termed 'Control and RTX' in Table 6.2). In 2005, the IETF sanctioned an experiment allowing IP/ECN on the SYN/ACK, termed ECN+.

The ECN++ proposal has found ways to safely allow each type of TCP control packet or retransmission to use IP/ECN, including FIN (finish) and RST (reset) packets as seen in the rightmost column of Table 6.2. ECN++ can be used with either standard ECN feedback or AccECN feedback. Except, the SYN can only be ECN-capable in the IP header if it requests AccECN feedback in the TCP flags (111). This is because only AccECN provides space in the SYN/ACK for feedback in the event that the SYN gets CE-marked.[17]

6.2 ECN++: Experiments

The goal of our experiments is to test how the ECN++ modifications to TCP/IP just described in Section6.1.4 would be treated in the current Internet, particularly over mobile networks. In order to do that, we designed two series of experiments. The first series of tests explores how the different ECN++ related fields are treated by the currently deployed base of network elements and servers. The second series of tests measures how the congestion control algorithms running in deployed servers react to ECN+ congestion signals. In both cases, we design our experiments to measure how equivalent non-ECN and ECN packets are treated in order to compare them with the

[17] If the server does not support AccECN at all, for safety the client has to behave as if it had received feedback of a CE on the SYN.

ECN++ measurement results.

6.2.1 ECN++ Support

As described in Section section 6.1 ECN++ enables the use of ECN in TCP control packets and pure ACKs. The experiments to test support for ECN++ exchange TCP control packets and pure ACKs with different values in the ECN-related fields and check how those packets are treated by the network and by servers. We next describe the tests performed for each type of packet.

TCP SYN and TCP SYN/ACK. To test support for ECN++ in the SYN and the SYN/ACK, we design the experiments to exchange packets containing the values used by ECN++ in the ECN field of the IP header and in the TCP ECN flags. We design two types of experiments, namely, client-side experiments and client/server experiments. In the client-side experiments, we only control the client side that sends the TCP SYN packets against existing servers in the Internet (Alexa top 100k servers). This allows us to learn information about a large number of paths and about support for ECN++ in both network elements and servers.

While client-side measurements provide a lot of information about support for ECN++ in the SYN packet, it usually provides little information about support for ECN++ in the SYN/ACK packet, since the SYN/ACK packet is generated by a server that is out of our control and does not support ECN++.

In the client/server experiments we control both the clients and servers of the connection. This allows us to perform exhaustive testing of all possible combinations of AccECN and ECN++ fields both in the SYN and the SYN/ACK. However, these experiments are limited to a few servers that we control, which also constrains the number of paths traversed.

Client-side Experiments. To observe how the ECN field in the IP header is treated by network elements we use Tracebox. [18] Tracebox uses the same principle as traceroute (i.e., sends packets with increasing TTL and receives an ICMP TTL exceeded error message from the router that discards the original packet when the TTL reaches zero). Tracebox uses information about the original packet returned in the ICMP error message to identify any changes in the IP header.

We run Tracebox sending TCP SYN packets with different codepoints in the ECN field of the IP header enumerated in Table 6.2. We executed these tests with all possible combinations of the CWR and ECE flags. With this test, we check whether the ECN field is modi-

[18] G. Detal, B. Hesmans, O. Bonaventure, Y. Vanaubel, and B. Donnet. Revealing Middlebox Interference with Tracebox. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 1–8, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1953-9. DOI: 10.1145/2504730.2504757

fied, and if so at which hop along the path it is modified.

While Tracebox is very powerful for seeing how the fields in the IP header are treated, it cannot detect the changes in the ECN flags in the TCP header. This limitation stems from the fact that most routers implement RFC792 [19] which requires them to return only the first 64 bits of the IP payload of the packet (leaving out all the ECN related flags later in the TCP header) while a few routers implement RFC1812 [20] which requires them to return the full packet if possible. Because of this, we implement a test that directly sends SYN packets from the clients we control to the Alexa top 100k servers with the different values for the ECN codepoints and TCP flags used by ECN and ECN++ as described in the rows corresponding to SYN packets in Table 6.2.

This test enables us to check, through the reception of the SYN/ACK, if the SYN was delivered to the server and the server processed it. We can further identify how many servers use RFC3168 [21] (Classic ECN) and how many servers use RFC5562. [22]

Client/Server Experiments. In these experiments we control the client and the server side. We implement both a client and a server side of the test that exchange every allowed ECN++ packet sequence. We also test for the case where ECN is not used. We test for the possible SYN-SYN/ACK packet sequences involving the different codepoint/flag combinations described in the SYN and SYN/ACK rows of Table 6.2.

Data packets, pure ACKs and FINs. We designed these experiments to show how the ECN++ FIN and pure ACK packets are treated by the network and the servers. We also measure how ECN-enabled data packets are treated to establish a baseline for comparison. Like previous experiments, we perform both client-side and client/server experiments (i.e., we perform these experiments with the Alexa top 100k servers and with our own servers).

In the experiments, the client uses Tracebox with PureACKs, Data packets, and FINs with the different combinations of the ECN codepoints and TCP flags included in the rows describing data packets, and 'Control and RTX' in Table 6.2. In all cases, the client establishes a standard ECN TCP connection before sending the test packets.

6.2.2 *Response to Congestion Signal*

We execute a number of client-side experiments to determine how the deployed base of ECN-enabled servers respond to ECN congestion signals. In particular, we want to learn if the congestion response

[19] J. Postel. Internet Control Message Protocol. RFC 792 (INTERNET STANDARD), Sept. 1981. URL http://www.ietf.org/rfc/rfc792.txt. Updated by RFCs 950, 4884, 6633, 6918, accessed on 2017-11-27

[20] F. Baker. Requirements for IP Version 4 Routers. RFC 1812 (Proposed Standard), June 1995. URL http://www.ietf.org/rfc/rfc1812.txt. Updated by RFCs 2644, 6633, accessed on 2017-11-27

[21] D. B. K. Ramakrishnan, S. Floyd. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3360 (Best Current Practice), Sept. 2001. URL https://tools.ietf.org/html/rfc3168. Accessed on 2019-02-02

[22] A. Kuzmanovic, A. Mondal, S. Floyd, and K. Ramakrishnan. Adding Explicit Congestion Notification (ECN) Capability to TCP's SYN/ACK Packets. RFC 5562 (Experimental), June 2009. URL http://www.ietf.org/rfc/rfc5562.txt. Accessed on 2017-11-27

to a CE marked packet echoed through one or more packets with the ECE flag set is equal to the response to three duplicate ACKs. We test for the case when a data packet is marked with CE (regular ECN) and the case when the CE marked packet is the SYN/ACK (ECN+ case). The approach we use is similar to the one used by TBIT. [23] For a given server, we measure the Initial Window (IW) of the TCP connection. More in detail, first we crawl the server's content to find a file that is large enough so that the TCP connection is not application limited in the downloading direction. Once we find such, we open a TCP connection to download it. We define the MSS to 100B, so that the Initial Window (IW) consumes as little as possible of the file[24]. The client sends the HTTP GET and does not sends any ACK back when packets arrive. Because the client is not sending any ACK back, the server will timeout and will retransmit the first data packet. We use the reception of the retransmission as a strong hint that the full IW has been sent by the server, so we calculate the IW as the number of bytes received since it first sent the GET.

After learning this, we establish a new TCP connection to the same server and we pretend that the first data packet sent by the server has experienced congestion (either pretending the packet is lost and by sending 3 duplicated ACKs or by sending the ACK for that packet with the ECE flag set) and we measure the resulting congestion window after such a congestion signal, learning the response to congestion from the server in these two situations (CE mark, or packet loss). In particular, we are able to learn if the response is the same for the two congestion signals. We also perform this test pretending that the packet that encountered congestion is the ACK of the TCP three way handshake (3WHS), allowing us to test the congestion response for servers that support ECN+.

6.2.3 *Experimental Setup*

We perform all the experiments we designed and described above between January and May 2017. For the client-side experiments, we use both the MONROE platform (mobile carriers) and PlanetLab (wired service providers). The MONROE platform [25] is the first open source [26] and open access hardware-based platform for independent, multihomed, large-scale experimentation in commercial mobile environments. MONROE allows authenticated external users to access the platform and deploy their own experiments. As explained in Chapter2 the platform comprises hundreds of nodes multihomed to three of the mobile providers in each of 4 EU countries (Spain, Italy, Sweden, Norway). For the purpose of this study, we instrument 11 MONROE nodes distributed in all the countries with MONROE

[23] J. Padhye and S. Floyd. On Inferring TCP Behavior. *Proc. ACM SIGCOMM'01, Computer Communication Review*, 31(4):287–298, Oct. 2001. (Aka. Identifying the TCP Behavior of Web Servers)

[24] RFC6928 defined the IW to be 10 MSS

[25] Ö. Alay, A. Lutu, R. García, M. Peón-Quirós, V. Mancuso, T. Hirsch, T. Dely, J. Werme, K. Evensen, A. Hansen, et al. Measuring and Assessing Mobile Broadband Networks with MONROE. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2016 IEEE 17th International Symposium on A*, pages 1–3. IEEE, 2016

[26] MONROE: Git repository, 2017. https://github.com/monroe-project/

coverage. We measure the following mobile carriers: Vodafone (IT), TIM (IT), WIND (IT), Orange (ES), Yoigo (ES), Movistar (ES), Telia (SE), Telenor (SE), Three (SE), Telia (NO), Telenor (NO). To test wired providers, we instrument 54 PlanetLab [27] nodes distributed in 25 networks over 22 countries.

All experiments test two TCP ports, namely port 80 and port 443. We execute all the tests both from the MONROE and Planetlab nodes. In all cases, we test only IPv4 hosts. We run the experiments we describe in Section6.2.1 and Section6.2.1 towards the Alexa top 100k web servers and to our own servers. We run the experiments we describe in Section6.2.2 towards the Alexa top 500k web servers, allowing us to also measure support for ECN in the wild. MONROE nodes resolve the Alexa top-N websites using Google's public DNS resolver; not the mobile carrier's default resolver. This enables us to build a vast and complex dataset, which we open to the community. For example, considering that each client attempts to send TCP SYN packets with 4 different codepoints in the ECN field of the IP header (Table 6.1) and TCP SYN and SYN/ACK packets with 5 different combinations for the TCP flags (Table 6.2), we collect information for a total of 26 million end-to-end communications, testing 6.5 million different paths.

6.3 ECN++: Good News for ++, Bad News for ECN over Mobile

In this Section, we describe the main findings of our measurement study.

Bad news: 7.5 out of 11 Mobile operators tested bleach ECN in outgoing packets. Using Tracebox in the MONROE nodes we find that 7 out of 11 Mobile operators bleach the ECN field in the IP header for all ECN codepoints in all packets going from the client to the Internet, for all types of packets tested (SYN, data packets, pure ACKS and FINs) and for both ports (80 and 443). The operators bleaching ECN are: Yoigo (ES), Orange (ES), Vodafone (IT), TIM (IT), Wind (IT), Telia (NO), Telia (SE). In all the bleaching cases, we observe that the ECN field is cleared in the first hop (i.e., the mobile operator is clearing it in its radio access).

One mobile operator (Movistar (ES)) bleaches the IP/ECN field only for port 80 and not for port 443. Using the client/server experiments, we learn that Movistar is using a web proxy that does not support ECN [28]. We find that two other providers, TIM and Vodafone, also use a proxy on port 80.

For the three mobile operators that honor the IP/ECN field in outgoing packets, (Telenor(NO), Telenor(SE) and 3 (SE)), we observe

[27] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman. PlanetLab: An Overlay Testbed for Broad-coverage Services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12, July 2003a. ISSN 0146-4833. DOI: 10.1145/956993.956995

[28] We observe that the TCP 3WHS is established between the client and the proxy first and when the HTTP GET is issued, the proxy establishes a connection with the server

that 0,53% of paths bleach the ECN field of outgoing packets further than 5 hops away. We find similar results in the experiments using the PlanetLab clients (0,23% bleaching), which is consistent with other measurements (see Chapter2).

Regarding incoming packets (towards the client), through client/server experiments we find that all providers that do not use a proxy honour the value in the ECN field. We verified this by sending ECT marked packets from our servers to all the MONROE clients in the different ISPs.

We conjecture that ECN bleaching could be due to a bug seen occasionally since 2009, where routers apply a DiffServ policy for bleaching the Differentiated Service Code Point (DSCP) that incorrectly wipes the ECN field as well (before 1998, both fields were combined in the ToS field). We try to validate that hypothesis by correlating changes to DSCP values and ECN bleaching. For this, we use Tracebox with all possible IP/ECT codepoints (see Table 6.1) and a random DCSP value. We find no correlation is possible because mobile operators always change the DSCP. These results neither invalidate nor confirm our hypothesis. We are in the process of contacting the ISPs to try to validate it.

Given the majority of mobile carriers that bleach the IP/ECN codepoint, we complement the set of mobile networks we measure with 7 additional operators active in 4 countries other than the ones with MONROE coverage (i.e., Elisa (FI), DNA (FI), Vodafone (DE), Blau (DE), Vodafone (UK), O2 (UK), Base (BE)). For these, we run a limited set of measurements only against four servers (two that we control and two that run in the wild). We find that out of the seven mobile carriers, 3 of them (DNA (FI), Vodafone (UK), O2 (UK)) present the same IP/ECN bleaching behavior in the first hop as the other 7.5 carriers we identified in MONROE.

Good news: ECN fields do not cause packet drop. As previously mentioned, Tracebox is unsuited for measuring traversal of the TCP/ECN flags because the deployed router base implements RFC792. In our dataset, we find that only 2,4% of routers (11k out of 468k) along the paths we tested are RFC1812-compliant. Through our client-side, end-to-end tests, we find that, irrespective of the TCP/ECN flags and IP/ECN codepoint combination, for the 6,5M paths tested, the packets sent using different TCP/ECN value combinations are acknowledged by the other end. We find this to be true for SYNs (we receive a SYN/ACK back), for data packets and FINs (we received ACKs back). It is true both for mobile and wired operators. This does not means the ECN related fields arrive at the other end unchanged though (we know it is not the case for the ECN field in the IP header, as per the previous finding). Through the client/server experiments, we observe that while the IP/ECN field is frequently bleached, all tested paths forward the TCP flags unchanged, except for the paths going through the mobile operators that use a non-ECN-capable proxy on port 80 (i.e., Movistar (ES) and TIM(IT)). Vodafone (IT) uses an ECN-capable proxy that honors the TCP flags.

Good news: ECN++ support is as good as ECN support. As described in the two earlier findings, we observe that in the 6,5M paths tested, ECN++ packets are treated by the network in the same way as ECN packets. In other words, ECN bleaching occurs as often in SYN, PureACKs and FIN packets as in data packets. The combinations of the ECN related fields used by ECN++ are not discarded or bleached more often than when used in ECN.

Good news: 61% of Alexa top 500k server supports ECN. We used the results of the experiments we describe in Section6.2.2, which we ran against Alexa top 500k web servers. Our finding corroborate the acceleration of ECN deployment reported in previous work. [29] We also observe that 3.51% of servers (10,709) support ECN in the SYN/ACK, a slight increase over the 1.3% reported in. [30] Five of them answer with ECT(1) in the SYN/ACK, while the others use ECT(0).

Good news: All the ECN-capable top-500k Alexa servers we were able to test have the same response to ECE as to 3 DupACKs. Out of the 305k Alexa servers that support ECN, we were able to test 158k of them for their congestion response to ECE marks. We found that their response to an ECE mark is the same to the response to 3 DupACKs. We were unable to test the congestion response of 147k servers for various reasons, including that we were unable to find enough content to fill the IW; or that they were redirected.

Good news: At least, 51% of the Alexa top-500k servers support IW of 10 segments. Additionally, out of the Alexa top-500k, 9.2% support IW of 2 segments and 9.3% support IW of 4 segments. We were unable to measure the IW of 74k (14%) of the servers because we were unable to find enough data to fill IW.

Bad News: 0.4% of the Alexa top-500k servers use IW larger than 10. The 0.4% we observe account for 1,745 servers, all using an IW larger than 10. Out of these, 1,121 servers deliver the whole file in the first RTT (the largest IW observed is 585 packets of 100 Bytes). Similar behavior was also previously reported in. [31]

Bad news: ECN+-enabled servers do not respond to congestion in the SYN/ACK. For the 3.51% of servers of the Alexa top-500K that respond with a SYN/ACK with the ECT codepoint, they all show the same odd behaviour, which superficially seems like ECN+.[32] However, none of them respond to an ECE flag in the ACK of the

[29] B. Trammell, M. K´uhlewind, D. Boppart, I. Learmonth, G. Fairhurst, and R. Scheffenegger. Enabling Internet-Wide Deployment of Explicit Congestion Notification. In *In Proc Passive & Active Measurement (PAM'15) Conference*, 2015

[30] B. Trammell, M. K´uhlewind, D. Boppart, I. Learmonth, G. Fairhurst, and R. Scheffenegger. Enabling Internet-Wide Deployment of Explicit Congestion Notification. In *In Proc Passive & Active Measurement (PAM'15) Conference*, 2015

[31] J. Padhye and S. Floyd. On Inferring TCP Behavior. *Proc. ACM SIGCOMM'01, Computer Communication Review*, 31(4):287–298, Oct. 2001. (Aka. Identifying the TCP Behavior of Web Servers)

[32] H. Schulzrinne. Location-to-URL Mapping Architecture and Framework. RFC 5582 (Informational), Sept. 2009. URL `http://www.ietf.org/rfc/rfc5582.txt`

3WHS, none respond with a second non-ECT SYN/ACK, and none reduce their initial congestion window, all contrary to the ECN+ RFC. Instead, they all enter Congestion Avoidance phase (i.e. in the second RTT the congestion window is 1 MSS larger than IW).

Part II Measuring NATs in the Wild

7 *Not all NATs are Equal: NATWatcher*

From a functional perspective, NATs only naturally supports client-server applications. The basic operation of a NAT relies on the creation of a mapping state between a private address and port pair and a public address and port pair. This state is created when a client, using a private address, initiates a communication with a server in the public Internet. Deploying applications that have an alternative paradigm, such as peer-to-peer applications, gaming or Voice-over-IP to name a few, that require hosts in the public Internet to initiate communications towards hosts behind a NAT is challenging and requires the use of the so-called NAT traversal techniques. These techniques are usually brittle and increase the latency, the traffic and the energy consumed by the endpoint. In this work we mine the obtained measurement results to identify common NAT profiles and usage in fixed and mobile lines, providing valuable data for application developers about the ground truth of NAT behavior in the current Internet for taking informing protocol design decisions.

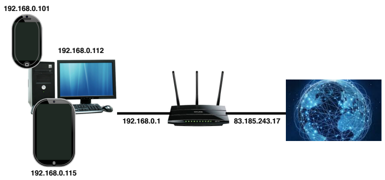


Figure 7.1: NAT operation.

7.1 NATs in Fixed Line

Network Address Translators (NATs) are now a commonplace in the Internet and they are included by default in the Internet Access offerings for both residential and mobile customers. Network Address Translators (NATs) were introduced back in the early 90s as a mean to cope with the incipient address depletion crisis. Together with Classless Interdomain Routing (CIDR), they successfully extended the lifetime of IPv4 from imminent depletion until very recently, when the Internet Assigned Numbers Authority (IANA) pool of IPv4 addresses finally ran out.

NATs successfully extended the lifetime of the IPv4 indeed, but at a high cost: they hardcoded the client-server paradigm in the architecture of the Internet.

While the aforementioned problem of supporting alternative application paradigms is fundamental to the nature of the NAT operation, it is exacerbated by the myriad of behaviors of the different NAT implementations deployed in the Internet. Different NATs use different criteria to create, preserve and remove their internal mapping state and have different filtering and forwarding rules. This severely complicates the job of applications willing to manage the NAT state in order to enable alternative communication models other than the client server one as they need to cope with all possible NAT flavours.

In order to achieve a more deterministic behavior from the NAT boxes, the Internet Engineering Task Force (IETF) produced a number of specifications defining the requirements that NATs should follow when creating, preserving and removing their internal state as well as some recommendations in terms of the different filtering and forwarding policies that NAT should implement. In particular, the IETF released NAT behavioral requirements for handling TTCP traffic [1], UDP traffic [2] and ICMP packets. [3] The goal of these requirements is to achieve a more deterministic behavior of NATs and hence significantly simplifying the job of deploying new application paradigms in the Internet, fostering innovation and competition. However, since these IETF standards specify the internal behavior for a NAT, it is far from trivial to assess whether NATs are following the recommendations and to the best of our knowledge there is no information about the prevalence of NAT boxes that honor the IETF specifications.

In this work, first we design and develop NATwatcher, a tool to measure key aspects of the behavior of the deployed NAT base, along with a measurement methodology based on crowdsourcing that allows us to perform large scale measurement using NATwatcher (§ 3).

7.1.1 IETF Standard for NATs

In this Section we describe all the requirements that the Internet Engineering Task Force (IETF) produced for NATs and specifically for CGNs. In the next chapters we explain the methodology we used for testing the following requirements.

TCP Requirements.

1. **TCP mapping.** NATs can be classified in 3 groups according to their mapping behavior:

    - Endpoint-Independent mapping: The NAT reuses the port mapping for subsequent packets sent from the same private IP address and port to any public IP address and port.

[1] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. IETF, RFC 5382, October 2008

[2] F. Audet and C. Jennings. Network Address Translation (NAT) Behavioral Requirements for Unicast UDP. IETF, RFC 4787, January 2007

[3] P. Srisuresh, B. Ford, S. Sivakumar, and S. Guha. NAT Behavioral Requirements for ICMP. IETF, RFC 5508, April 2009

- Address-Dependent mapping: The NAT reuses the port mapping for subsequent packets sent from the same private IP address and port to the same public IP address, regardless of the external port.
- Address and Port-Dependent mapping: The NAT reuses the port mapping for subsequent packets sent from the same private IP address and port to the same public IP address and port while the mapping is still active.

2. **IP address pooling and persistent.** Some applications require that same source IP address is used for the entire session. The NAT should have an IP address pooling behavior of Paired. This means that the same internal IP address should be mapped to the same external IP for the entire duration of the session. The NAT can have also an IP address persistent behavior (independently from the IP pooling behavior). The IP address can be preserved over time.

3. **Supporting the Simultaneous Open Connection Establishment.** Simultaneous connections are used when 2 hosts want to reach each other. In this case there is no a 3 way handshake, but 2 simultaneous connections (2 way handshake): each host sends a SYN and receive the SYN of the other host, ACKs it and then wait for its ACK. A NAT sometimes blocks the inbound SYN and blocks or translate badly the outbound SYN/ACK.

4. **TCP filtering**. When an unsolicited packet is received from the Internet, the NAT applies filtering rules that can be classified as follows:

   - Endpoint-Independent filtering: The NAT forwards any packets destined to an internal host as long as a mapping exists.
   - Address-Dependent filtering: The NAT filters packets received from a specific external host to a specific internal host if the internal host has not previously sent any packet to that specific external host's IP address.
   - Address and Port-Dependent filtering: The NAT filters packets received from a specific external host to a specific internal host if the internal host has not previously sent any packet to that specific external host's IP address and port.

5. **Unsolicited inbound SYN packets.** A NAT MUST NOT respond to an unsolicited inbound SYN packet for at least 6 seconds after the packet is received. If during this interval the NAT receives and translates an outbound SYN for the connection the NAT MUST

silently drop the original unsolicited inbound SYN packet. Otherwise, the NAT SHOULD send an ICMP Port Unreachable error (Type 3, Code 3) for the original SYN. As well as verifying the requirements of 6 seconds, we measure the time for the inbound SYN packet response.

6. **Source port selection.** The NAT leaves the port unchanged when creating a mapping and implement port parity, if so an even port will be mapped to an even port, and an odd port will be mapped to an odd port. This behavior is recommended from the RFCs. Some applications expect that at least the port range is maintained.

7. **Supporting hairpinning.** Hairpinning enables a host in the home network to access another host in the home network using the external IP address. As well as verifying the support of hairpinning, we verify the "External source IP address and port" for hairpinning.

8. **Behaviour with respect to the ICMP Destination Unreachable.** The NAT should translate also this message.

9. **Mapping and ICMP packets.** The mapping should be maintained after that an ICMP Destination Unreachable packet is received by the NAT.

10. **TCP Connectivity with the STUN Server.** The NAT should allow TCP STUN requests.

11. **Packet modification.** A NAT should maintain the packets it has been sent.

UDP Requirements.

1. **UDP mapping.**

2. **Source port selection.**

3. **Mapping lifetime over 2 minutes.** NATs are required to maintain UDP mappings during no less than 2 minutes.

4. **Outbound refresh behavior.** According to the UDP requirements, outgoing packets must refresh the existing binding.

5. **UDP filtering.**

6. **Supporting hairpinning.**

7. **Deterministic behavior.** The NAT maintains the same mapping or filtering during time.

8. **Unreachable ICMP packets.** The mapping should be maintained after that an ICMP Destination Unreachable packet is received by the NAT.

9. **Mapping and ICMP packets.** Receipt of any sort of ICMP message MUST NOT terminate the NAT mapping.

10. **Supporting the "do not fragment" flag.** The NAT should properly generate the "ICMP fragmentation needed" message when a packet is received with the "do not fragment" flag set and discard that packet.

11. **Out-of-order UDP fragments.** When packets are fragmented and received out of order some NATs are required to try to reassemble and then forward the packet.

ICMP Requirements.

1. **ICMP query from internal.**

2. **ICMP query session less over 60 sec.** An ICMP Query session timer MUST NOT expire in less than 60 seconds.

3. **Handling ICMP error packet.** When an ICMP Error packet is received, if the ICMP checksum fails to validate, the NAT SHOULD silently drop the ICMP Error packet.

4. **Refreshing mapping ICMP error packet.** If a NAT device receives an ICMP Error packet from the external realm, and the NAT device does not have an active mapping for the embedded payload, the NAT SHOULD silently drop the ICMP Error packet. If the NAT has active mapping for the embedded payload, then the NAT MUST do the following prior to forwarding the packet

5. **Error packet hairpinning.** NATs are required to support hairpinning for error messages.

6. **Support of ICMP destination unreachable.**

7. **ICMP time exceeded.**

8. **Timestamp and Timestamp Reply Messages.**

9. **Source route options.** A NAT device MAY support modifying IP addresses in the source route option so the IP addresses in the source route option are realm relevant. If a NAT device does not support forwarding packets with the source route option, the NAT device SHOULD NOT forward outbound ICMP messages that contain the source route option in the outer or inner IP header.

This is because such messages could reveal private IP addresses to the external realm.

10. **Address Mask Request/Reply Message.**

11. **Parameter Problem Message.**

12. **Non-QueryError ICMP Messages.** A NAT MAY drop or appropriately handle Non-QueryError ICMP messages.

7.2 NATwatcher: Profiling NATs in the Wild

In order to test the aforementioned NATs behavior.

- First, we design and develop NATwatcher, a tool to measure key aspects of the behavior of the deployed NAT base, along with a measurement methodology based on crowdsourcing that allows us to perform large scale measurement using NATwatcher.
- Second, using the proposed methodology, we perform a large measurement campaign using the methodology explained before.
- Finally, we mine the obtained measurement results to identify common NAT profiles, providing valuable data for application developers about the ground truth of NAT behavior in the current Internet.

Similarly to the methodology used in Section4.3.1, we use Microworkers platform to recruit workers around the world that are using a NAT to connect to the Internet to run out NATwatcher tool in order to characterize the behavior of their respective NAT.

7.2.1 NATwatcher: Methodology Overview

NATwatcher is the tool we build to detect the characteristics of a NAT using a number of active tests. Figure 7.2 summarizes the operational setup of NATwatcher. In a nutshell, the worker downloads and executes the NATwatcher application. The NATwatcher application automatically executes the tests to characterize the NAT behavior, by sending different combinations of packets to our Measurement server deployed in the public Internet and processes the response packets sent from the server back to the NATwatcher client (step 1 in Figure 7.2) . Once all the tests are completed the application sends the compiled measurement results to a collector server where they are stored (step 2 in Figure 7.2) and the code that can be used to redeem the payment.

We designed NATwatcher to be suitable to be used in a crowdsourcing environment. In particular, we designed it to be as simple as

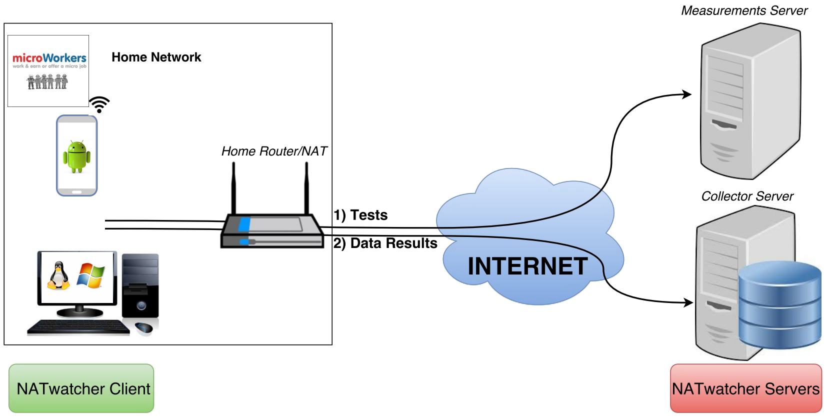


Figure 7.2: NATwatcher operational setup.

possible in order to be tractable for a large number of workers. Workers are not required to have any technical background. Workers just need to download and execute the NATwatcher application and all data are automatically collected and reported to our collector server without further worker intervention.

In order to reach a high number of workers, we developed three versions of NATwatcher, one for Android, one for Windows and one for Linux. We therefore create two crowdsourcing campaigns, *Android-based* and *Windows/Linux-based*, which we detail next.

Android-based Campaign — This campaign requires the worker to install our Android application. The worker then just have to launch the application and all the tests are then run sequentially. In order to ensure that workers' Internet access is provided through a fixed line (and not 3G, 4G, etc.), the application is instrumented to run the *microjob* using the WiFi interface.

Windows and Linux-based Campaign — This campaign requires workers to download the application and run it from a Windows or Linux machine. The application takes care of all the measurements to be performed as in the Android app. We asked workers to perform the test from their own personal computers, using a fixed line.

We also deploy a measurement server, connected to the public Internet. This server implements the server side of the tests described below including the UDP tests, the UDP STUN tests and the TCP STUN tests.

7.2.2 *NATwatcher Tests*

According to our experience, a *microjob* offered in a crowdsourcing platform is less likely to be performed by a large number of workers if it takes more than 5 minutes. In order to minimize of execution time of the *microjob*, we carefully selected 17 key NAT characteristics that NATwatcher will test for. We use the tests described in. [4] as starting point We then expand the tests set and adapt them to be run in a crowdsourcing platform. We developed all the tests in C programming language to be able to reuse the code through the different platforms (i.e., Android, Windows and Linux). All the tests are implemented crafting UDP, ICMP and TCP packets using Raw Sockets or Standard Sockets whenever possible.

TCP Tests. For the first four tests enumerated below, we use the same methodology than for UDP tests, but using TCP packets.

1. **Mapping behavior**: The test verifies if the NAT is Endpoint-Independent, Address-Dependent or Address and Port-Dependent with respect to the TCP mapping behavior.

2. **Filtering behavior**: This test verifies the filtering behavior of the NAT with respect to TCP packets (Endpoint-Independent, Address-Dependent and Address and Port-Dependent filtering).

3. **Port preservation**.

4. **Hairpinning support**: The test checks the support of hairpinning for TCP packets.

5. **Mapping and ICMP packets**: This test verifies if the mapping is maintained after that an ICMP Destination Unreachable packet is received by the NAT.

UDP Tests.

1. **Mapping behavior**: The test verifies if the NAT assigns the same mapping for communications between a specific internal IP address and port and any external IP address and port for UDP packets. NATs can be classified in 3 groups according to their mapping behavior:

   - **Endpoint-Independent mapping**: The NAT reuses the port mapping for subsequent packets sent from the same private IP address and port to any public IP address and port.
   - **Address-Dependent mapping**: The NAT reuses the port mapping for subsequent packets sent from the same private IP address and port to the same public IP address, regardless of the external port.
   - **Address and Port-Dependent mapping**: The NAT reuses the port mapping for subsequent packets sent from the same private IP address and port to the same public IP address and port while the mapping is still active.

[4] C. Jennings. Nat Classification Test Results. IETF Internet Draft draft-jennings-behave-test-results-04, July 2007

The test is as follows: we first send two STUN *binding requests*[5] to two different public addresses of our STUN server[6]. We compare the address and port returned in the two STUN responses received. If both the addresses and the ports match then we conclude that the mapping behavior of the NAT is *Endpoint-Independent*. If this is not the case, we send a third binding request to our STUN server using the primary addresses used before and a different port. If the address and port reported in the STUN response is the same than the one reported before when using the primary address and a different port, the NAT is *Address dependent*, else the NAT is *Address and Port-Dependent*.

2. **Filtering behavior**: The test detects the filtering behavior of the NAT. When an unsolicited packet is received from the Internet, the NAT applies filtering rules that can be classified as follows:

   - **Endpoint-Independent filtering**: The NAT forwards any packets destined to an internal host as long as a mapping exists.
   - **Address-Dependent filtering**: The NAT filters packets received from a specific external host to a specific internal host if the internal host has not previously sent any packet to that specific external host's IP address.
   - **Address and Port-Dependent filtering**: The NAT filters packets received from a specific external host to a specific internal host if the internal host has not previously sent any packet to that specific external host's IP address and port.

   The test is as follows: We send a binding request to the primary address of our STUN server with a *change port* and *change address* attributes. These binding request attributes solicit the server to send the response from the alternate IP address and port. If the client receives the response, then the filtering behavior of the NAT is *Endpoint-Independent*. If not, we send a third binding request to the primary address with only *change port*. If the client receives a response then the NAT is *Address-Dependent Filtering*, if not it is *Port-Dependent Filtering*.

3. **Port Preservation**: This test checks if the NAT leaves the port unchanged when creating a mapping. We send a packet using a particular local source port and then we compare the port number with the external bound port. If they match the NAT is performing port preservation. If they do not match, it is possible that the NAT does not implement port preservation, but it is also possible that the specific port used as source port was already in use in another mapping (from another internal host). Because of this limitation,

[5] The STUN protocol operates as follows: the client located behind the NAT sends a binding request message to a STUN server. The STUN server replies with a response message containing the IP address and port of the client in its payload, as observed from the server.

[6] A STUN server has two public IP addresses, the primary one and the alternate one.

this test only provides a lower bound to the number of NATs that implement port preservation. It would be possible to increase the accuracy of this test by repeating the test several times. However, this would increase the time budget of the test, which we cannot afford.

4. **Hairpinning support**: Hairpinning enables a host in the home network to access another host in the home network using the external IP address. In order to check it we send two packets: the first one to discover our mapped address using STUN protocol and the second one from a different source port and towards the discovered mapped address. If we receive the response it means the NAT supports hairpinning.

5. **Supporting the "do not fragment" flag**: the NAT should properly generate the "ICMP fragmentation needed" message when a packet is received with the "do not fragment" flag set and discard that packet. Our application sends a packet with the Do not Fragment flag set and waits for the reception of the correspondent ICMP message.

6. **Out-of-order UDP fragments**: When packets are fragmented and received out of order some NATs are required to try to reassemble and then forward the packet. In order to check this behavior, we send from our server disordered fragments and check their reception in the application.

7. **Mapping lifetime over 2 minutes**: NATs are required to maintain UDP mappings during no less than 2 minutes. In order to check this, we send a first binding request to our server and after 2 minutes we send another binding request using a different port but specifying that the response should be sent to the previous binding. If the answer is not filtered, it means that the previous mapping has been preserved.

8. **Outbound refresh behavior**: According to the UDP requirements, outgoing packets must refresh the existing binding. This test is similar to the previous one, except that an additional outgoing packet is sent one minute after the initial binding request and the second binding request is sent after 3 minutes.

ICMP tests.

1. **Reply/Request of ICMP packets**: The test verifies if the NAT supports simple ICMP Reply/Request message exchange by sending a query and waiting for the answer.

2. **Supporting of ICMP Destination Unreachable**: This test verifies the NAT is capable of forwarding ICMP Destination Unreachable messages generated as a response to a previous UDP packet. We send an UDP packet to our server and the server replies with Destination Unreachable behavior. We then verify is the ICMP error is received by the NATwatcher client.

3. **Supporting of ICMP Time Exceeded**: Same as the previous one but for ICMP Time Exceeded messages.

4. **Supporting error packet hairpinning**: NATs are required to support hairpinning for error messages. NATwatcher check this by sending two packets, the first one to discover our mapped address using STUN protocol and the second one is an ICMP Echo Error message sent to the mapped address from a different port. The NATwatcher verifies the reception of the error message.

We implement all the tests described before in the Linux and Windows versions of NATwatcher. However, 9 of the 17 tests need root permissions for their execution and unfortunately, Android devices are not commonly rooted. For this reason we implement only 8 over 17 NATwatcher tests in the case of the Android app. Specifically we implement test number 1, 2, 4, 7 and 8 for UDP NAT behavior and tests number 1, 2 and 4 for TCP behavior.

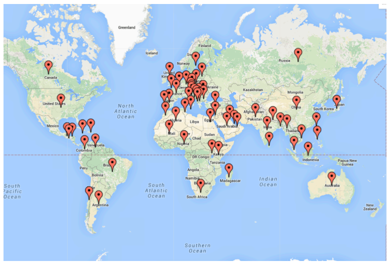


Figure 7.3: Worldwide distribution of vantage points.

7.3 *NATwatcher: Results and Dataset*

In this Section, we present the collected data set, and we analyze the obtained results.

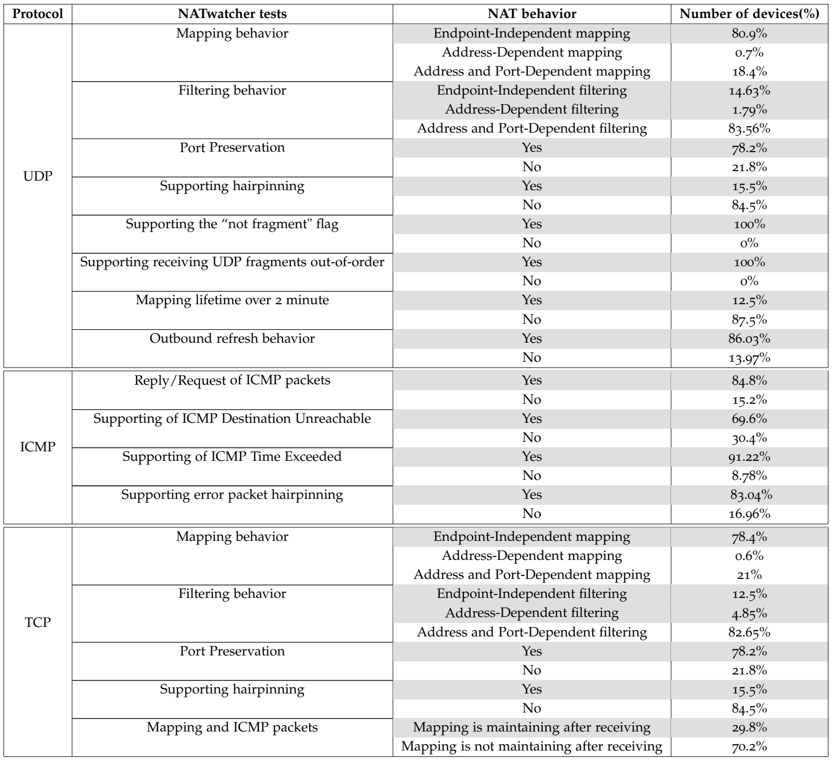
7.3.1 *Crowdsourcing Campaign*

We defined two campaigns, one for the Windows/Linux-based application (each worker attempts 17 tests) and the other one for Android (each worker attempts 8 tests) and both were available for 28 days during June 2016. Overall, we recruited 781 workers: 170 workers participated in the Windows/Linux-based campaign, while 611 workers participated in the Android-based one.

After the campaigns, our data-set consists of 7,778 unique tests results.

Overall, workers are distributed among 65 countries (cf. Figure 7.3). The devices cover more than 280 ISPs for a total of 120 different OUIs. [7]

[7] We use the WHOIS database to retrieve the ISP from the public IP address of each NAT and we use the MAC address of the NAT to retrieve the OUI. Since some vendors use multiple OUIs or rebrand other products, we clarify that when we refer to 120 different vendors in the paper we refer to 120 different OUIs.



Protocol	NATwatcher tests	NAT behavior	Number of devices(%)
UDP	Mapping behavior	Endpoint-Independent mapping	80.9%
		Address-Dependent mapping	0.7%
		Address and Port-Dependent mapping	18.4%
	Filtering behavior	Endpoint-Independent filtering	14.63%
		Address-Dependent filtering	1.79%
		Address and Port-Dependent filtering	83.56%
	Port Preservation	Yes	78.2%
		No	21.8%
	Supporting hairpinning	Yes	15.5%
		No	84.5%
	Supporting the "not fragment" flag	Yes	100%
		No	0%
	Supporting receiving UDP fragments out-of-order	Yes	100%
		No	0%
	Mapping lifetime over 2 minute	Yes	12.5%
		No	87.5%
	Outbound refresh behavior	Yes	86.03%
		No	13.97%
ICMP	Reply/Request of ICMP packets	Yes	84.8%
		No	15.2%
	Supporting of ICMP Destination Unreachable	Yes	69.6%
		No	30.4%
	Supporting of ICMP Time Exceeded	Yes	91.22%
		No	8.78%
	Supporting error packet hairpinning	Yes	83.04%
		No	16.96%
TCP	Mapping behavior	Endpoint-Independent mapping	78.4%
		Address-Dependent mapping	0.6%
		Address and Port-Dependent mapping	21%
	Filtering behavior	Endpoint-Independent filtering	12.5%
		Address-Dependent filtering	4.85%
		Address and Port-Dependent filtering	82.65%
	Port Preservation	Yes	78.2%
		No	21.8%
	Supporting hairpinning	Yes	15.5%
		No	84.5%
	Mapping and ICMP packets	Mapping is maintaining after receiving	29.8%
		Mapping is not maintaining after receiving	70.2%

Table 7.1: IETF NAT requirements compliance results, based on the NATwatcher test.

7.3.2 *General Results*

Table 7.1 summarizes the results for both campaigns. For each test we show the percentage of NATs that follow a particular behavior. We color in grey the IETF recommended behaviors.

We can see that for 11 out of 17 there is wide compliance with the IETF sanctioned behavior (for these 11 tests, more than 80% of the analyzed NATs follow the recommended behavior). For the remaining 6 tests, the large majority of tests NAT boxes do not follow the IETF recommendation.

Most of the measured NATs models implement UDP and TCP Endpoint-Independent mapping, support UDP not fragment flag and receiving UDP fragments out-of-order, moreover most of them follow all the ICMP requirements. All these requirements are reported as a MUST in [8], [9] and [10]. For the rest of the tests the success rate decreases dramatically. In particular it is interesting to see less than 16% of the devices fulfill the requirements on UDP and TCP Endpoint-Independent filtering, hairpinning support and mapping lifetime over 2 minutes. The Endpoint-Independent filtering is not a mandatory requirement, but it is highly recommended if transparency is a priority. If security is the priority, NAT should follow an Address-Dependent filtering, but as results show, the majority of NATs set an

[8] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. IETF, RFC 5382, October 2008

[9] F. Audet and C. Jennings. Network Address Translation (NAT) Behavioral Requirements for Unicast UDP. IETF, RFC 4787, January 2007

[10] P. Srisuresh, B. Ford, S. Sivakumar, and S. Guha. NAT Behavioral Requirements for ICMP. IETF, RFC 5508, April 2009

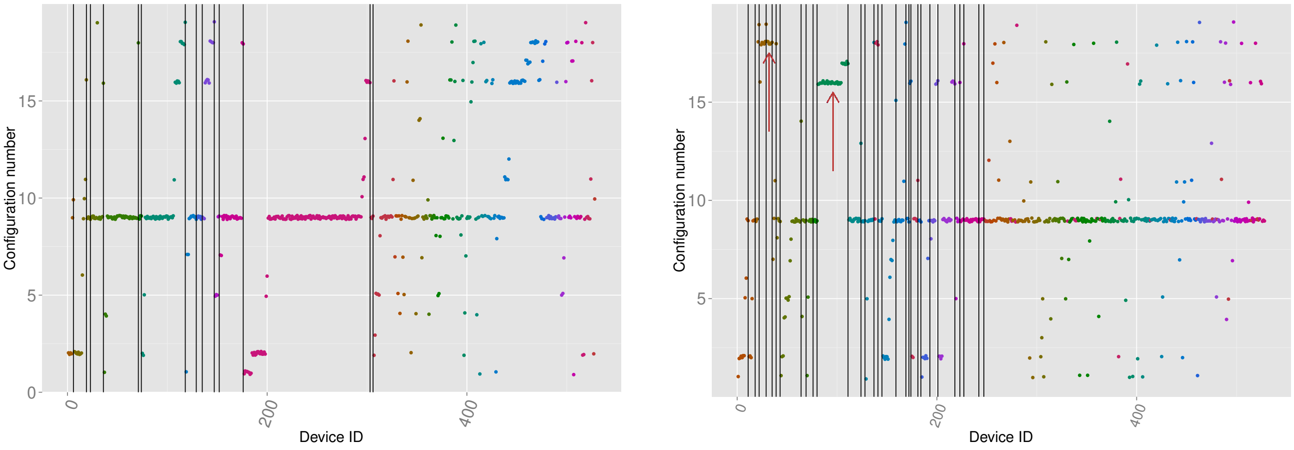
Address and Port-Dependent filtering. Applications such as online gaming for instance would not benefit from this kind of behavior.

Hairpinning and binding lifetime over 120 seconds are two mandatory requirements for UDP. Most of the NATs do not support these features as the failure rate is higher than 84% for these tests.

Port preservation support is recommended particularly for outgoing TCP connections, in order to allow NAT port prediction. In our results, we find a lower bound of 78% of devices that use port preservation.

Results show that about 70% of NATs delete the TCP mapping if an ICMP error message is received affecting that specific mapping. This is strongly recommended against in [11] as it exposes the NAT to attacks relying on ICMP error messages to delete existing NAT bindings.

7.3.3 *NAT Classification*



a) Devices ID in order of vendors

b) Devices ID in order of ISPs

Figure 7.4: Configuration number vs. device ID.

In this subsection we identify the most common NAT configurations. For this analysis we only consider the 8 of the tests have been performed for all the tested NAT devices (i.e. the 8 tests that are executed in all the campaigns). We characterize a configuration through the different combinations of the results of these 8 tests. We identified 19 different configurations across the 781 NATs we tested. We detail in Table7.2 the configurations that appear in more than 5 devices (11 out of 19).

We assign the number to each configuration basing on the IETF requirement compliance (i.e., we named “1" the configuration with higher RFC requirements compliance and the one with less RFC requirements compliance is the number “19"). For example, configu-

[11] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. IETF, RFC 5382, October 2008

	NAT Tests								
Configuration Number	UDP mapping	UDP filtering	UDP hairpinning	TCP mapping	TCP filtering	UDP mapping over 2 min	UDP outbound refresh	TCP hairpinning	**Number of devices (%)**
9	E.I.	A.P.D.		E.I.	A.P.D.		✓		52.5
2	E.I.	E.I.	✓	A.P.D.	A.P.D.		✓	✓	5.5
18	A.P.D.	A.P.D.		E.I.	A.P.D.				5.2
5	E.I.	E.I.		A.P.D.	A.P.D.				2.5
1	E.I.	E.I.	✓	E.I.	E.I.		✓	✓	2
11	E.I.	A.P.D.		A.P.D.	A.P.D.				1.8
17	A.P.D.	A.P.D.		A.P.D.	A.P.D.		✓		1.5
4	E.I.	E.I.		E.I.	E.I.				1.15
7	E.I.	A.D.		E.I.	A.D.				0.8
19	A.P.D.	A.P.D.		E.I.	A.D.		✓		0.8
8	E.I.	A.P.D.		E.I.	A.P.D.		✓		0.65

Table 7.2: Most common NAT configurations. For each configuration it is reported the behavior with respect to each test. An Endpoint-Independent behavior is reported as E.I., A.D. indicates Address-Dependent and A.P.D. describes an Address and Port-Dependent behavior. For other tests RFC requirements compliance are shown with a tick.

ration number 1 in Table II is the configuration recommended in the different RFCs, which has UDP Endpoint-Independent mapping and filtering, TCP Endpoint-Independent mapping, supports hairpinning and the UDP mapping lifetime is over 2 minutes. We consider this configuration “open", meaning that transparency is the top priority compared to security.

Overall 52.5% of devices implement the configuration number 9. The devices come from over 50 countries and 173 unique ISPs, for a total of 72 different vendors. The most common configuration includes Endpoint-Independent mapping for UDP and TCP and the filtering as strict as possible (Address and port dependent).

This is a clear trend towards security for the rest of configurations, as only 4 over 11 NAT configurations implement Endpoint-Independent filtering behavior.

Table II also shows that for both UDP mapping lifetime and the hairpinning, the majority of analyzed devices fail to comply with the IETF recommendations.

7.3.4 *Vendors and ISPs*

In this Section we try to figure out if the adoption of a certain common NAT configuration depends on the vendor or on the ISP.

Figure 7.4 shows the configuration number for each tested device. The devices (in the horizontal axis) are ordered first by vendor (Figure 7.4 (a)) and second by ISP (Figure 7.4 (b)). Each vendor/ISP is delimited by a vertical line. Devices from vendors/ISPs present in our data set with not more than 2 devices are grouped after the last line on the right.

Apart from a clear trend to use configuration number 9, as mentioned earlier, we observe some "open" configurations for few vantage points in some vendors. This can be explained considering that users can change the basic configuration that vendors impose by default to accommodate the NAT to their own needs (i.e., activating hairpinning or Endpoint-Independent filter).

In Figure 7.4 (b), we identify 2 ISPs (identified by arrows) where the behavior for all NATs connected to these ISPs is very uniform, exhibiting one or two configurations. These two ISPs are mobile ISPs that provide 3G and LTE Internet access services. Given that NATwatcher only runs using the WiFi interface and not the cellular one, this means that the NATwatcher client was executed in a mobile phone connected to a hotspot which in turn was connected to the Internet using 3G or LTE. We believe it is safe to conclude that in these cases, the NATwatcher client was behind two cascaded NATs, the one from the WiFi access point and the one from the mobile operator. This means that when the client executes NATwatcher, the results reflect the superposition of both NATs along the path, reflecting the more restrictive behavior of the two, which is likely to the one of the NAT of the mobile operator explaining the uniform behavior observed.

8 Tracking and Characterizing CGNs in Fixed Lines Europe and U.S.

In light of the IPv4 address scarcity problem, another approach towards prolonging the life of current IPv4 address allocations is to deploy Carrier Grade NATs (CGNs), where Internet Service Providers (ISPs) share the same public IPv4 address across multiple end users. CGNs may introduce a number of issues for end users, service providers, content providers and government authorities. [1] There is some evidence that CGNs can cause dropped services in peer-to-peer applications, and lead to low performance of file transfer and video streaming sessions. [2] CGNs also introduce security challenges including traceability of IP addresses and anti-spoofing. Despite these challenges, CGNs offer an immediate relief to the IPv4 address scarcity problem, so it is likely that their popularity will increase over time.

8.1 CGNs in Fixed Line

The use case for CGN differs in wireline vs. mobile networks. Given the rapid boost in the number of operational Internet-enabled mobile devices, and the scarcity of available IPv4 address space, mobile operators usually assign the same public IP address to multiple end users. Hence, some form of CGN technology has always been the norm, rather than the exception. [3] [4] In wireline networks, however, end users are typically assigned public IPv4 addresses. While this situation may change as IPv4 addresses become increasingly scarce and ISP customer bases continue to grow, there is no source of systematic measurements of the prevalence or evolution of CGN deployments in ISPs.

In order to achieve a more deterministic behavior from the CGNs, the Internet Engineering Task Force (IETF) produced a number of specifications defining the requirements that NATs should follow when creating, preserving and removing their internal state as well

[1] M. Ford, M. Boucadair, A. Durand, P. Levis, and P. Roberts. Issues with IP Address Sharing. RFC 6269, June 2011

[2] C. Donley, L. Howard, V. Kuarsingh, J. Berg, and J. Doshi. Assessing the Impact of Carrier-Grade NAT on Network Applications. RFC 7021, September 2013

[3] Z. Wang, Z. Qian, Q. Xu, Z. Mao, and M. Zhang. An Untold Story of Middleboxes in Cellular Networks. In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, pages 374–385, 2011. ISBN 978-1-4503-0797-0

[4] S. Triukose, S. Ardon, A. Mahanti, and A. Seth. Geolocating IP Addresses in Cellular Data Networks. In *Passive and Active Measurement*, pages 158–167. Springer, 2012

as some recommendations in terms of the different filtering and forwarding policies that CGNs should implement. [5] However, it is far from clear to what extent the currently deployed CGN base complies with the aforementioned requirements. Learning the behaviour of currently deployed CGNs is an important input for application and protocol designers, so that they can adapt their design to work properly in the Internet.

There is also the concern that CGNs utilization may result in performance penalties. It is a common operational practice that a single CGN serves a very large number of mobile nodes (up to millions of devices per CGN). Because of this, the CGN itself may become a bottleneck and impose performance penalties to the communication. This is especially so for the case of the first packet that creates the NAT mapping, but it may also affect traffic forwarding once the mapping has been established. Measuring and understanding the performance penalties resulting from the use of CGNs is important for application designers, for end users and for mobile operators adopting the CGN technology. (e.g. for application designers, this information should affect the maximum number of communications open in parallel). This input is very relevant for a measurement platform such as MONROE, since the performance penalties if exist can affect and bias the measurement results.

8.1.1 CGNs Usage in a Famous ISP in Italy

During our studies on CGNs we were very keen on learning about operational CGN solutions in the Internet, deployment architectures and CGN configurations. We contacted various ISPs and we got some information about the usage of CGNs on an ISP in Italy:

- The reason for deploying a CGN solution is the IPv4 depletion.
- The customer of the IP Ratio is 1 to 8.
- They deploy CGN at each ISP PoP, they have actually deployed about 20 PoP.
- The estimation of the size of the IPv4 address pool they configure at each CGN is typically /20.
- For each CGN they assign Min 32768 – Max 65536 users.

8.2 Tracking the Big NAT across Europe and the U.S: Revelio

We perform a large scale active measurement campaign to detect CGNs in fixed broadband networks using NAT Revelio. [6] Revelio

[5] S. Perreault, I. Yamagata, S. Miyakawa, A. Nakagawa, and H. Ashida. Common Requirements for Carrier-Grade NATs (CGNs). RFC 6888, April 2013

[6] A. Lutu, M. Bagnulo, A. Dhamdhere, and K. Claffy. NAT Revelio: Detecting NAT444 in the ISP. In *International Conference on Passive and Active Network Measurement*, pages 149–161. Springer, 2016

enables us to actively determine from within residential networks the type of CGN. Our contributions are twofold. First, we performed a large measurement campaign to detect CGN in fixed broadband access networks. Second, as a result of this large-scale measurement campaign, we added additional tests to Revelio to decrease the probability of false positives when quantifying the degree of CGN penetration in broadband providers.

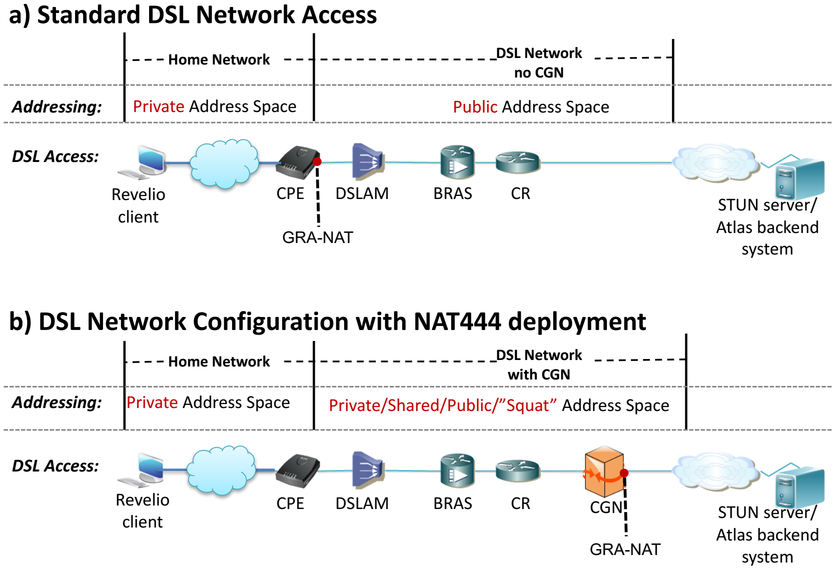


Figure 8.1: *Revelio experimental setup for a DSL access network (the DSLAM, BRAS and CR are standard elements in the DSL architecture). The Revelio client runs on a device connected to the home network, whose exact topology we do not know. In the case of (a) standard DSL network access, the CPE performs the translation to the GRA, thus it is the GRA-NAT. In the case of (b) standard DSL network configuration with NAT444 deployment, the CGN is the one performing the translation to the GRA, thus it is the GRA-NAT.*

In Figure 10.4 we depict the residential setup we consider for Revelio in the context of a DSL access network. The home network may have an arbitrary topology consisting of multiple hosts, routers and switches including multiple levels of NATs. The home network connects to the Internet through the Customer Premises Equipment (CPE) also known as “home router" or “home gateway". The access link connects the CPE with the ISP access network. In the case of DSL technology, the access network includes the digital subscriber line access multiplexer (DSLAM), the broadband remote access server (BRAS) and the Core Router (CR). The ISP network connects with the rest of the Internet.

In terms of IP addressing, the home network generally uses private IP address space. The home gateway usually performs the NAT function (home-NAT) from the private addresses within the home network to the addresses used in the ISP access network which may be public, private or shared. [7] In some cases, end-users can configure several different realms of private addresses within their home network in the context of cascaded home NATs. Independently of the home network topology, when a host within the home network com-

[7] J. Weil, V. Kuarsingh, C. Donley, C. Liljenstolpe, and M. Azinger. IANA-Reserved IPv4 Prefix for Shared Address Space. RFC 6598, April 2012

municates with a host in the rest of the Internet, the private address used by the host in the home network translates to a public address that we call the Globally Routable Address (GRA). For the majority of the residential Internet market, the ISP configures the GRA on the Internet-facing interface of the CPE and the NAT function in the CPE translates from the private addresses in the home network to the GRA. [8] An alternative, incipient, setup is one including an additional NAT function that operates in the ISP network (in addition to the NAT function in the CPE) and performs the final translation to the GRA. These configurations are usually called Carrier Grade NAT (CGN), Large Scale NAT (LSN) or NAT444. In this case, packets flowing between the home network and the Internet go through two upstream NAT-capable devices: the CPE (customer grade NAT) and the ISP NAT (Carrier-Grade NAT). The goal of the Revelio methodology is detect CGNs by distinguishing whether the NAT function translating to the GRA is located within the home network or it is located in the ISP network.

In order to discern where the translation to the GRA occurs, Revelio performs active tests from a device connected to the home network. The *probe* running Revelio connects to the home network and may or may not be directly connected to the CPE, i.e., there may be multiple hops, including ones performing NAT function(s), between the *probe* and the CPE. The target of the active tests the *probe* performs are servers located in the Internet (Figure 10.4). NAT Revelio does not require any cooperation from the ISP beyond forwarding Internet packets to and from the customer.

8.2.1 Revelio Methodology Overview

As mentioned earlier, the purpose of Revelio is to detect whether the *device performing the translation to the GRA* (hereinafter, the GRA-NAT) *resides in the home network or in the ISP network*. In order to do this, Revelio attempts to pinpoint the location of the GRA-NAT with respect to the access link. If the GRA-NAT lies between the *probe* and the CPE, we conclude that the user is not behind a CGN. If the GRA-NAT lies after the CPE, we conclude that the ISP deploys CGN. In order to achieve this, Revelio needs to determine the location of the GRA-NAT and the location of the access link with respect to the *probe* and compare them.

Initial Revelio tests. To determine the location of the GRA-NAT, Revelio performs the following steps: 1) discovers the GRA by running STUN [9] against a public STUN Server located in the Internet (FCC MBA) or by using the Atlas API (RIPE Atlas); 2) runs traceroute

[8] S. M. Garland and D. B. Smith. Communications Between Service Providers and Customer Premises Equipment, Dec. 26 2000. US Patent 6,167,042

[9] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing. Session Traversal Utilities for NAT (STUN). RFC 5389, October 2008a

to the GRA, computing the number of hops from the probe to the GRA-NAT;

The determination of the location of the access link is challenging because we aim to support arbitrary topologies in the home network and we do not have any prior information about where in the home network the *probe* connects. To determine the location of the access link we make the following assumption: the propagation delay of the access link is one order of magnitude or higher than the propagation delay of the links in the home network. We believe this is a realistic assumption for the different access technologies and home network technologies available in the market and it is supported by existing empirical evidence. [10] In order to locate the access link, we estimate the propagation delay of different links between the *probe* and an arbitrary target server in the Internet using *pathchar*. [11] *Pathchar* is a well-known technique for estimating transmission and propagation delays along the path using multiple traceroute measurements with different packet sizes. Since we only need to estimate the order of magnitude of the delays, the precision of *pathchar* is sufficient. Using the propagation delays measured by *pathchar*, we determine the access link as the first one with a propagation delay at least one order of magnitude higher than the previous links.

By comparing the respective locations obtained for the GRA-NAT and for the access link, we can establish whether the GRA-NAT is located *before* the access link (no CGN) or *after* the access link (ISP uses CGN).

Supplementary Revelio tests. In addition to the main detection tests, the Revelio client performs two other tests to gain additional insight about the presence of CGNs. In particular, Revelio performs the following tests:

Invoke UPnP actions. If the *probe* directly connects to the CPE (i.e., the access link is one hop away), Revelio tries to run the UPnP protocol [12] from the *probe* to retrieve the IP address of the WAN-facing interface of the CPE. If this IP address matches the GRA, we conclude that there is no CGN. Otherwise, if the UPnP query returns a private/shared address, Revelio detects an upstream CGN.

Private/shared addresses along the path: To detect the access link location, Revelio runs multiple traceroute measurements to a fixed target. This enables us to retrieve the IP addresses operators configure in their network. We then search for private and shared addresses after the access link. The detection of private/shared addresses after the access link alone does not imply the presence of an upstream CGN, but it serves as a hint that the ISP might be operating one. In particular, the presence of shared addresses after the access

[10] S. Sundaresan, W. De Donato, N. Feamster, R. Teixeira, S. Crawford, and A. Pescapè. Broadband Internet Performance: a View from the Gateway. In *ACM SIGCOMM Computer Communication Review*, volume 41, pages 134–145. ACM, 2011

[11] A. B. Downey. Using Pathchar to Estimate Internet Link Characteristics. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '99, 1999

[12] UPnP Forum. UPnP Specifications. `http://upnp.org/sdcps-and-certification/standards/`. Accessed: 2016-06-17

link provides a stronger indication about the presence of CGNs, because this address block is specifically reserved for CGN deployment [13]

8.2.2 Evolving Revelio

After the first phase of the large scale measurement campaign (May 2016) with Revelio on RIPE Atlas and FCC MBA and communicating with several of the ISPs we measure, we identified some corner cases that may confuse Revelio, which we describe in detail below. The common link between these corner cases is the fact that they counter the logic of the access link detection approach we integrated in the original Revelio test-suite. Given Revelio's reliance on the correct detection of the access link location, incorrectly mapping non-standard home network topologies leads to false positives. To tackle these particular issues and increase the robustness of Revelio, we enhance the original methodology by adding the following two tests to the test-suite.

Pathchar to the GRA. We identified 165 home gateways that replied to traceroute to the GRA as if they were two hops (148 in RIPE Atlas and 17 in FCC MBA), thus generating spurious links in the results. When processing the traceroute to the GRA from the *probe*, the home gateways generate one reply from the internal interface (for packets with TTL=n) and a second reply from the external WAN-facing interface that assigns the GRA (for packets with TTL=n+1). When running pathchar to the external server, however, the home gateway only replies to traceroute from the internal interface (for packets with TTL=n), while the subsequent reply comes from the following hop (for packets with TTL=n+1). This behavior breaks the Revelio methodology because the GRA appears to be one hop farther from the *probe* than it actually is, making us believe that the GRA-NAT is past the access link. We detected this behaviour in several models of home routers, including SpeedPort or FritzBox.

In order to control for these cases, we run *pathchar* from the *probe* to the GRA, similarly to the pathchar to the external server. The home gateway replies as if it were two hops, generating a spurious link that masquerades as the access link. We contrast the access link propagation delay value we obtained running *pathchar* to the external server with the one we obtained running *pathchar* to the GRA and observe that they are significantly different. If the CPE generates two replies to the traceroute to the GRA, the delay of the spurious link from the CPE to the GRA measured by the *pathchar* to the GRA is significantly smaller than the delay of the *real* access link measured by the *pathchar* to the external server. This is so because the spurious

[13] J. Weil, V. Kuarsingh, C. Donley, C. Liljenstolpe, and M. Azinger. IANA-Reserved IPv4 Prefix for Shared Address Space. RFC 6598, April 2012

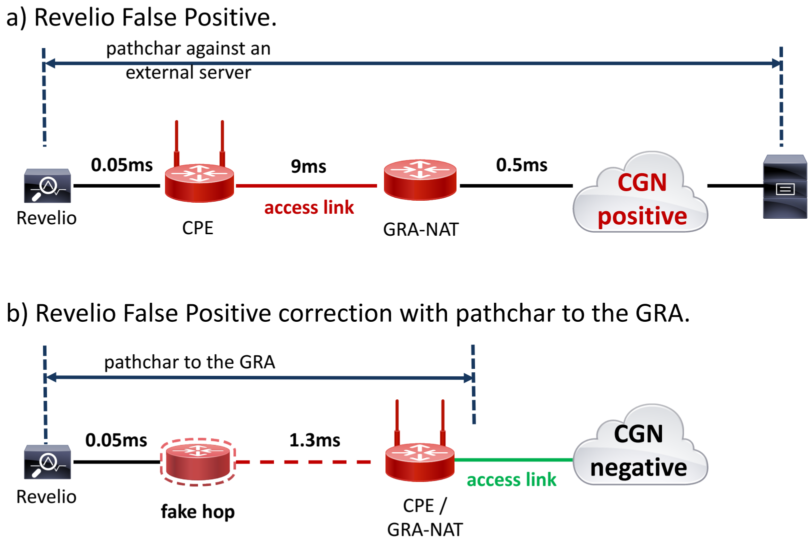


Figure 8.2: *We observed that some home gateways reply to traceroute as if they are two hosts, generating spurious links. When running pathchar to an external server, Revelio shows that the access link is the spurious link (Figure (a)). We run pathchar to the GRA (Figure (b)) and compare the results with the previous output from pathchar to the external server to correct these Revelio false positives.*

link is internal to the CPE, while the link we measure with *pathchar* to the external server is the actual access link. By comparing the two *pathchar* results, we can identify these anomalous CPEs and correctly identify the GRA at hop n.

We exemplify this non-standard behavior in Figure 8.2. The original Revelio suite generated positive results for upstream CGN because the GRA-NAT appears to be outside the home network (Figure 8.2(a)) based on the *pathchar* to the external server test and the number of hops until the GRA (from traceroute to the GRA). The *pathchar* to the external server results shows that the access link is the red link with a delay of 9ms. In the same time, we see that the GRA-NAT is after the access link, in the access network, based on the results of the traceroute to the GRA. However, when running *pathchar* to the GRA, we obtain a different value for the propagation delay on what seems to be access link (Figure 8.2(b)). The dotted red link is actually not the access link, but the spurious link that is internal to the CPE. Given the large difference between the value of the propagation delay on the spurious link (Figure 8.2(b))) and the propagation delay on the actual access link (Figure 8.2(a)), we conclude that the home router replied to the traceroute to the GRA as two hops and Revelio generated a false positive.

Expected access technology delay. In some cases, we have detected that the propagation delay of the different links within the home network differs in one order of magnitude (e.g., one link with a delay of tens of μs and another one with delay in the hundreds of μs), confusing the Revelio methodology. In this case, the delays of both home network links are still one order of magnitude less than the propagation delay of the actual access link. In order to deal with this case, we define an expected range for the access link delay based on the access technology and we verify if the access link delay we measure falls within the expected range. If this is not the case, we mark the first link that falls within the expected range as the access link. In the case of the FCC MBA platform, we know the access link from the per-probe metadata we received. For RIPE Atlas, we leverage the user tags that also contain information about the access technology.

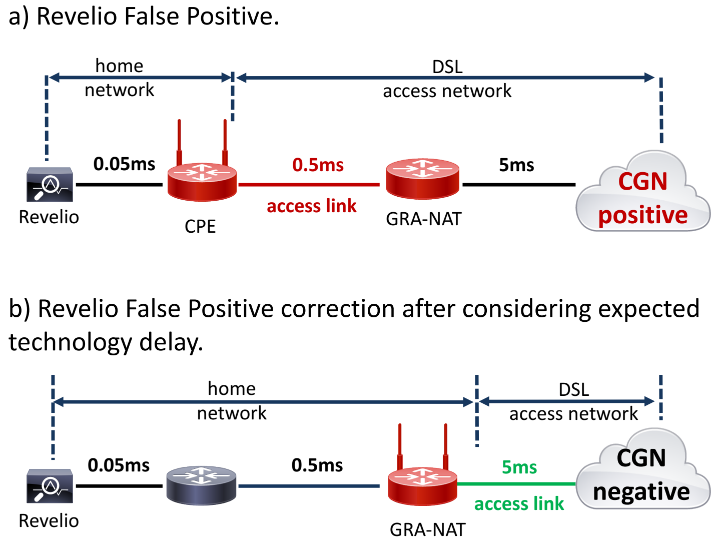


Figure 8.3: *We observed that propagation delay of the different links within the home network differs in one order of magnitude. When applying the original Revelio methodology, we erroneously conclude that the GRA-NAT is in access network of the ISP (Figure (a)). When factoring the expected propagation delay corresponding to the access technology we correct the Revelio false positive (Figure (b)).*

We exemplify such an example in Figure 8.3. We first observe that the two red routers in Figure 8.3.(a) reply to the traceroute probing with private addresses. Then, after running pathchar we obtain the propagation delay for the links. The first link with a propagation delay higher by one order of magnitude than the previous one is the one we mark as the access link. Because we detect that the GRA-NAT is after the access link (and we also detect private addresses in the access network) we conclude that there is an upstream CGN active. However, when comparing the propagation delay on the access link with the expected propagation delay corresponding to DSL access (i.e., between 2ms and 30ms), we conclude that the first two links are part of the home network, and the access link is the green link in

Figure 8.3.(b).

In Figure 8.4 we show the spread of access link propagation delay values we obtained using the original Revelio methodology. The horizontal lines show the expected range of propagation delay per access technology. The points that fall outside the expected range are potential false positives which we tackle within the evolved Revelio suite.

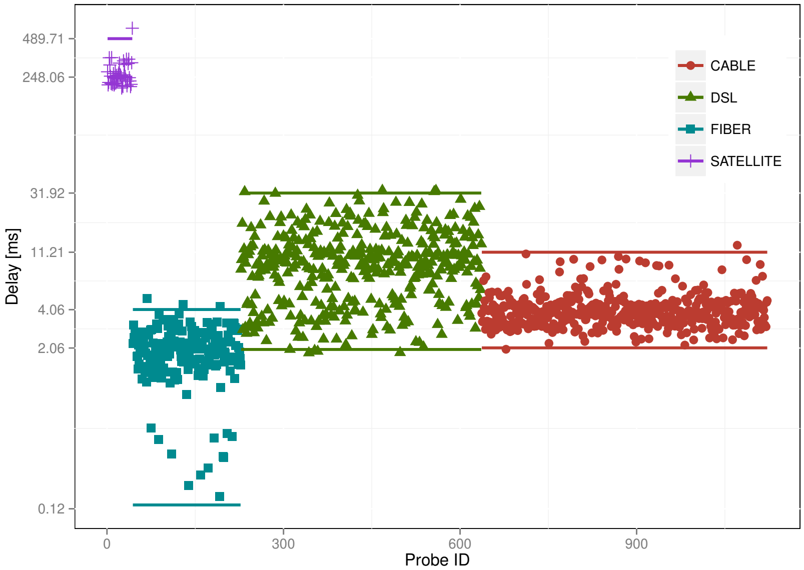


Figure 8.4: *Scatter plot of the access link delays we identified using the original Revelio methodology. The horizontal lines represent the range od expected access link propagation delay per access technology. The points that lie outside the expected range are potential false positives in the Revelio results.*

8.2.3 *Revelio Limitations*

The evolved Revelio methodology still suffers from a series of limitations, which we discuss in this Section.

The Revelio methodology heavily relies on traceroute. If traceroute packets are filtered, the Revelio tests will be inconclusive because we cannot reliably locate the access link. However, in the case of traceroute to the GRA, receiving ICMP error replies from at least one hop immediately after the access link (i.e., after the CPE in Figure 10.4) is enough to establish whether the GRA-NAT lies after the access link, in the ISP network. In other words, even if the traceroute does not reach the target GRA, the Revelio methodology only requires for the packet to be routed outside the home network, in the ISP network.

Anecdotal evidence exists about home routers rate-limiting generation of ICMP packets [14], which can hinder the *pathchar* methodology we use to measure the link propagation delays. Rate-limited ICMP errors will add additional delay to the traceroute messages,

[14] T. Burbridge. Personal Communication, 2016

thus artificially increasing the delay we estimate for that link. Since we subtract ICMP round-trip delays to infer per-link delay, ICMP rate-limiting will mislead our inference. Also, the lack of information about the type of access technology may increase the number of false positives in the case of non-standard home topologies. However, this is a non-issue when the end-user deploys Revelio to verify her upstream configuration. In the case of deploying Revelio on other hardware platforms or crowdsourcing the Revelio measurements, we assume this type of information is not hard to obtain from the platform operators.

One of the Revelio tests requires the *probe* to send UPnP queries to the CPE, which should also support UPnP. These devices do not always support UPnP, limiting the effectiveness of Revelio. The Atlas probes do not allow users to perform UPnP queries. Additionally, for some of the SK probes, the CPEs do not support UPnP and cannot reply to the query the *probe* sends.

8.3 Revelio Deployments and Data

We deployed the evolved Revelio test-suite in two large measurement platforms, the FCC-MBA platform and RIPE Atlas. We ran the measurement campaign in two phases (May 2016 and August 2016) on both platforms. Based on the experimental results from the first phase (May 2016), we evolved the methodology of NAT Revelio to tackle a number of corner cases that confused our original approach. We enhanced the test suite to account for a wide diversity of home network topologies and various access technologies. In the second phase of the measurement campaign (August 2016) we deployed the evolved Revelio suite to investigate the state of CGN deployment in broadband networks.

We next detail the two platforms and describe the dataset we obtained.

FCC-MBA platform: The United States Federal Communication Commission’s (FCC) initiative “Measuring Broadband America" program [15] is an ongoing nationwide performance study of broadband service in the United States that aims to improve the availability of information for consumers about their broadband service. The initiative manages a large hardware-based measurement platform operated by SamKnows (SK), an international statistics and analytics firm supporting similar projects in other countries around the world. The *SK probes* are off-the-shelf routers that users voluntarily host in their home networks. The *SK probes* run pre-installed software that measures Internet connection performance metrics including download speed, upload speed or latency. Apart from the standard

[15] Measuring Broadband America. https://www.fcc.gov/general/measuring-broadband-america

pre-installed measurements, the *SK probes* can execute other custom tests. We deployed the NAT Revelio client to run on 2,477 *SK probes* operating as part of the MBA testbed. This allowed us the unique convenience of running *active measurements from inside the end-users home networks* across the U.S. and attempt to reveal the presence on a NAT in their ISP. For each probe in the FCC-MBA platform, we also have access to metadata regarding the access technology and ISP of each user line, which SamKnows collects and maintains. Based on this information, the FCC-MBA panel covers 23 different ISPs and tests 4 different access technologies overall. These include 10 major DSL providers, 8 different cable providers, 3 ISPs that offer fiber to the home Internet connectivity and 2 main satellite providers in the U.S. In terms of aggregated number of *probes* per technology, according to the metadata we have available, cable Internet access is predominant with approximately 40% of the *probes* operating on cable access. DSL follows with a share of approximately 20% of the *probes* we tested. Fibre access lines account for 9% of the US-Revelio testbed. 3.5% of the testbed is made up of satellite access lines.

RIPE Atlas platform: For the RIPE Atlas probes, we use the API to check the user tags regarding the access technology, which we retrieve from the information that the hosts of the probes volunteer to RIPE. At the time we deployed the Revelio suite, Atlas consisted of 9,231 active probes distributed in 3,381 Autonomous Systems (ASes), according to the RIPE Atlas API.

To ensure a statistically significant sample of vantage points in an ISP, we only tested the ISPs with at least 20 Atlas probes. We aggregated the probes per AS using the probe metadata. Thus, we deployed Revelio in 2,644 *Atlas probes*, corresponding to 43 different ASes. However, the set of tests that we were able to run in the *Atlas probes* is more limited than the ones we can run on the *SK probes*. In particular, the *Atlas probes* do not provide support for UPnP. Thus, as opposed to FCC-MBA where we ran the complete Revelio test suite, on the Atlas infrastructure we only deployed the traceroute-based tests.

8.3.1 Revelio Dataset

We scheduled the Revelio client to run over 20 times on each *probe* during August 2016 for the FCC-MBA platform and in the RIPE Atlas platform. The data we collected from the *probes* in *each* run of Revelio are the following: the Globally Routable Address (GRA), the mapped port number, traceroute results to the GRA, traceroute results to a fixed target address (with 21 different packet sizes), UPnP query result to retrieve the IP address on the external interface of the device to which the *probes* connect (only for *SK probes*). In total, we collected data from 5,121 *probes* in 64 ISPs with an average of 20 repetitions[2] per *probe* which resulted in over 2 million traceroutes. The ISPs we tested include 42 DSL providers, 16 cable providers, 4 ISPs that offer fiber to the home Internet connectivity and 2 satellite providers. We are in the process of releasing the FCC-MBA Revelio dataset after proper anonymization. The RIPE Atlas measurements are publicly available and retrievable using the RIPE Atlas API. We can make available the measurement identifiers (MIDs) corresponding to Revelio tests upon request (interested parties can retrieve them to then query the RIPE Atlas database).

We processed the raw data we collect using Revelio and combined it with metadata we received from SamKnows and RIPE to build the *Revelio State* for each *probe*. The Revelio State consists of the device identifier, ISP name, access technology, local IP address of the *probe*, GRA, number of hops we measure until the GRA, location of the access link, set of private IPs we detect *immediately* after the CPE (if any), set of shared IP addresses we detect after the access link (if any) and number of times Revelio successfully ran on the *probe*. Revelio then feeds this information to the algorithm for detecting the type of upstream NAT.

8.3.2 *Catching the Big NAT*

We deploy NAT Revelio to detect the presence of CGN on a total of 5,121 different customer lines. In function of the upstream NAT configuration, Revelio classifies each *probe* into one of the following cases: (i) *inconclusive* (cases Revelio was unable to draw any conclusion due to incomplete or inconsistent results). (ii) *no home NAT* (i.e., the *probe* where Revelio runs is directly connected to the public Internet), (iii) *simple home NAT* (the CPE performs the GRA-NAT), (iv) *Carrier Grade NAT* (the GRA-NAT is outside the home network, in the ISP's network) and We aggregate the results by the inferred upstream NAT configuration (Table 8.1).

Inconclusive. For 1,276 *probes* (307 *SK probes* and 969 *Atlas probes*), Revelio gave inconclusive results either because none of the tests could run on the *probe* or because we did not obtain enough information to properly interpret the results we were able to collect. Our approach is conservative and tags as inconclusive the case of mixed responses from different tests. For example, traceroute limitations and ICMP traffic being filtered along the path to the external target server hamper our capacity to identify the access link (Section 8.2.3). Without knowing the location of the access link, when the end-user deploys several levels of NAT in the home, we cannot draw conclu-

[2] In the FCC-MBA platform, in order not to interfere with normal user Internet activity, the *probes* perform cross-traffic sensing and run the tests we schedule only when they detect no end-user traffic. Thus, the number of Revelio repetitions differs for various measurement vantage points.

ISP ID	CC	Tech.	# of probes	Inconclusive	Simple Home NAT	Carrier Grade NAT	Confirmed
1 (Undisclosed ISP)	US	Satellite	76	0	0	76	Yes
2 (Kabel Deutschland)	DE	Cable	49	27	14	8	Partially
3 (Fastweb)	IT	Fiber	26	14	8	4	Yes
4 (OTE)	GR	DSL	21	5	14	2	No Reply
5 (Liberty Global)	NL	Cable	280	133	146	1	Yes
6 (Zen)	UK	DSL	32	11	20	1	No Reply

Table 8.1: **Revelio Positive Results:** List of ISPs with at least one probe with ***positive*** Revelio result (i.e., operates behind a CGN). We report the Country Code (CC), the access technology (Tech.), the total number of probes we tested for that ISP (# of probes), the number of probes for which Revelio gave inconclusive results (Inconclusive), the number of probes Revelio tested negative (Simple Home NAT), the number of probes Revelio tested as positive (Carrier Grade NAT) and the current status of the confirmation from representatives of the ISP with positive Revelio results (Confirmed). For the latter, we mark this field with *Yes* if the ISP confirmed the Revelio results at the IP level, *Partially* if the ISP confirmed they use CGN but did not confirm the specific IP lines tested, *No Reply* if we did not get any feedback from the ISP.

sions regarding the presence of NAT in the ISP. These *probes* account for approximately 24% of the total, (12% of the *SK probes* and 36% of the *Atlas probes*). We discard these cases from further analysis.

No home NAT. Revelio found that in 299 different cases (85 in *SK probes* and 214 *Atlas probes*), the Revelio client was running on a *probe* configured with a public IP address that was also the GRA. These *probes* were operating in the public Internet, which implies that the lines were not connected behind a NAT solution. In all these cases, the *traceroute to the GRA* test also confirmed the lack of a NAT solution in the corresponding ISPs.

Simple home NAT. Out of the rest, for 3,454 *probes* (2,009 *SK probes* and 1,445 *Atlas probes*) Revelio established the presence of simple home NAT and excluded the possibility of further NAT in the ISP. Revelio reports the simple home NAT configuration (and, thus, the lack of NAT in the ISP for the respective line) when at least one of the *traceroute to GRA* and *invoking UPnP actions* tests establish that the home gateway is performing the GRA-NAT. In the case of the UPnP test, for 1,300 *SK probes* the address retrieved through UPnP from the CPE matched the GRA, concluding that the CPE was the GRA-NAT. For 815 *SK probes*, the Revelio client was unable to communicate with the CPE through UPnP, either because the CPE did not supported UPnP or because the *SK probe* was not directly connected to the CPE. In the case of the traceroute to the GRA test, for 2,965 *probes* (1,520 *SK probes* and 1,445 *Atlas probes*) Revelio located the GRA-NAT before the access link, concluding that the CPE was also the GRA-NAT. As a interesting data point, using *pathchar to the GRA* test Revelio purged 165 of cases where the CPE replied as being two different hops, creating false positives. In particular, Revelio detected this behavior in one single ISP for 78 out of 228 *probes*.

Carrier Grade NAT. For 92 *probes* in 6 ISPs (76 *SK probes* in 1 ISP and 16 *Atlas probes* in 5 ISPs) Revelio detected the presence of CGN technology in the ISP's network. Table 8.1 details the number of *probes* that tested positive for CGN per ISP[3] We identified one satel-

[3] We only disclose the names of the ISPs we tested using the RIPE Atlas platform. We are currently pending the approval of the FCC for disclosing the names of the ISPs we tested with the FCC-MBA testbed.

lite provider in the U.S. where all *probes* tested positive for CGN. For the rest of the ISPs, we detected a mix of some *probes* that tested positive for CGN and others that did not. Overall, about 2% of the *probes* tested positive for CGN. About 10% of the ISPs we tested hosted at least one *probe* that tested positive for CGN. Of these latter ones, only one ISP had a widespread deployment of CGN, while the other ISPs presented a few scattered *probes* that tested positive, hinting a localized deployment, e.g., possibly for trials or suggesting a specific service.

8.3.3 *Validation of Revelio Results*

NAT Revelio tested 5,121 Internet lines in 64 different ISPs worldwide. In total, it reported 92 end users with an upstream CGN, which connected to 6 different ISPs. We validated both the positive (upstream CGN) and negative (no upstream CGN) results at the IP level through different means, including direct contacts with the involved ISPs or, in one case, using the WHOIS database information.

Positive Revelio Results. We obtained confirmations at the IP level from 4 ISPs (89 *probes*) for the presence of CGN in their network for the lines we tested and received no replies from the other 2 ISPs (3 *probes*). In Table 8.1 we report on the status on communication with the ISPs for which Revelio identified the presence of CGN. In particular, for ISP#1 from Table 8.1 – the satellite provider in the US for which all *probes* tested positive – the operator confirmed that its normal configuration includes performing the NAT function in the ISP network and that all the 76 lines that tested positive were indeed behind a CGN. ISP#3 (Fastweb) confirmed both the positive and the negative Revelio results. For ISP#5 (Liberty Global) from Table 8.1, the GRA associated with the *probe* is actually tagged in the WHOIS database (in the *Organization* field) as CGNAT (the other 279 *probes* in the same ISP did not have a GRA in the subnet marked as CGN). ISP#2 (Kabel Deutchland) from Table 8.1 confirmed that it is using CGN in its network. However, we did not obtain explicit confirmation from their representatives that the exact lines we detected as positive are actually behind a CGN, which is why we marked it as a *partial* confirmation.

Based on the ground truth we collected, we conclude that NAT Revelio did not generate any false positives. Thus, provided that NAT Revelio can successfully run (see Section 8.2.3 for limitations), its precision[4] is 100% reported to the set of probes which the ISPs validated.

Negative Revelio Results. Out of the 5,121 lines Revelio tested, its results pointed to a simple NAT configuration (no CGN) for 3,454

[4] The precision represents the ratio between the number of true positives and the sum of the true positives and the false positives.

probes in 63 different ISPs. For the negative results, we obtained validation from 4 ISPs for which all *probes* tested negative for upstream CGN in the ISP. The 4 ISPs account for 508 *probes*. We mention that (confirmed) negative results from Revelio testing do not preclude the existence of CGN technology in the corresponding networks.

Based on the ground truth we collected, we conclude that NAT Revelio did not generate any false negatives. Thus, provided that NAT Revelio can successfully run, its recall[5] is 100% reported to the set of probes which the ISPs validated. However, the Revelio methodology reported inconclusive results in 24% of the cases (this numbers drops to 12% if the measurement platform supports both UPnP and traceroute based tests). This is a consequence of Revelio's limitations, which we detail in Section 8.2.3.

8.3.4 *Analysis of Results*

The information we retrieved through the Revelio tests reveals additional insight about confirmed CGN operational setups, which we discuss next.

Number of hops between the CPE and the CGN. In 76 of the 92 positive cases, the CGN was located in the first IP hop in the ISP network. This is the case for all the 76 cases of the US satellite ISP that tested positive and as well for ISP#6 (Zen). The other ISPs had 2 to 6 IP hops between the CPE and the CGN. This reflects two different CGN deployment architectures. The ISPs that exhibit only one hop between the CPE and the CGN deploy the CGN functionality in the first IP aggregation point in the network (e.g., the BRAS). This is consistent, for example, with some CGN cards that are available for insertion in the BRAS products. The other deployment setup installs the CGN in the ISP core network, allowing the aggregation of a higher number of costumers in a single CGN box. This enables a more efficient multiplexing of the public IPv4 address pool.

Addressing. In terms of addressing used in the hops between the CPE and the CGN, ISP#1 and ISP#3 (Fastweb) assign shared address space [16] to devices in the access network. Interestingly, ISP#1 uses shared address in the modem interface facing the home network and uses public IP address for all hops in the ISP network. ISP#3 uses both shared addresses and private addresses (the first hop after the CPE uses shared address space and the subsequent ones private address space). The remaining ISPs use a mix of public and private addressing between the CPE and the CGNs.

GRA stability over time. We analyzed the number of GRAs that we identified for each probe during the period we performed the tests. We found that the majority of the *probes* (4,388 probes, ac-

[5] The recall represents the ratio between the number of true positives and the sum of the true positives and the false negatives.

[16] J. Weil, V. Kuarsingh, C. Donley, C. Liljenstolpe, and M. Azinger. IANA-Reserved IPv4 Prefix for Shared Address Space. RFC 6598, April 2012

counting for 85% of the total) have only one GRA during the whole tested period. The remaining 733 *probes* use 2 to 19 different GRAs in the analyzed period. The average number of GRAs per *probe* for all the *probes* is 1.3 while the average number of GRAs per *probe* for the *probes* that tested positive for CGN is 3. In particular, for ISP#1 the mean number of GRAs goes up to 4. We can see that while frequently changing the GRA is by no means exclusive of the CPEs behind a CGN, CGN deployments exhibits a considerably higher number of GRAs per customer. Additionally, we searched for GRAs simultaneously used by multiple *probes*/CPEs. We found two GRAs simultaneously used in ISP#1, one for 2 hours and the other for 48 hours. (Incidentally, we found 40 GRAs that were used by 2 *probes* each, but we discovered that these cases were home networks that were hosting multiple Atlas probes. This is likely related to the incentives for hosting probes in Atlas).

9 Tracking and Characterizing Carrier Grade NATs in Mobile Broadband Networks in Europe

Mobile Broadband Networks operators have deployed Carrier Grade NATs (CGNs) as the only realistic mean to accommodate the ever-increasing number of mobile nodes connected to their networks. While CGNs enable mobile operators to provide Internet-access to millions of devices with a limited amount of IPv4 addresses, their adoption imposes functional and potentially also performance penalties to the end-users that may affect their perceived quality of experience.

While the difficulties in supporting alternative application paradigms are fundamental to the nature of the NAT operation, the functional limitations resulting from CGN adoption are exacerbated by the myriad of behaviors of the different NAT implementations deployed in the Internet.

9.1 CGNs in Mobile Broadband Networks

In this Chapter we describe CGNWatcher, a measurement tool that executes a number of active tests to fully characterize CGN deployments in MNOs. The CGNWatcher tool systematically tests more than 30 behavioural requirements of NATs defined by the Internet Engineering Task Force (IETF) and also multiple CGN behavioural metrics. We have deployed CGNWatcher in MONROE and performed large measurement campaigns to characterize the real CGN deployments of the MNOs serving the MONROE nodes. The tool was designed to easily allow future features to easily be integrated in MONROE.

The measurement results obtained through CGNWatcher are relevant to application and protocol developers to inform their design about how to overcome the limitations imposed by CGNs. The information retrieved through CGNWatcher are also useful for experimenters using MONROE, as CGNs may have an important impact in the feasibility of experiments and can potentially bias the results if not accounted for.

We deploy our experiment considering three fundamental tasks:

- Design, development and implementation of CGNWatcher.
- Execution of the measurements on MONROE.
- Analysis of the measurement results.

9.1.1 IETF Standard for CGNs

We design active metrics to characterize CGNs on the MONROE MNOs. Particularly, we define:

TCP Requirements.

1. **TCP mapping.** We first send two STUN binding requests to two different public addresses of our STUN server. We compare the address and port returned in the two STUN responses received. If both the addresses and the ports match then we conclude that the mapping behavior of the NAT is Endpoint-Independent. If this is not the case, we send a third binding request to our STUN server using the primary addresses used before and a different port. If the address and port reported in the STUN response is the same than the one reported before when using the primary address and a different port, the NAT is Address dependent, else the NAT is Address and Port-Dependent. Additionally we tests the behavior of the NAT when we send packets with different source ports, but same IP address. We check wherever we get the same public IP address.

2. **IP address pooling and persistent.** We sends a SYN from a host we control to the CGNwatcher client and sniff the response in the client. We then sends a SYN/ACK from the CGNwatcher client and sniff the response in the server. We can also apply the TCP state tracking from MAO paper SIGCOMM: two tests.

   - SYN-out SYN-in tests: if a NAT allows an incoming SYN packet after an outgoing SYN.
   - SYN-out SYN-ACK-out tests: if a NAT allows a client to send out a SYN-ACK packet after sending a SYN packet.

3. **TCP filtering**. We send a binding request to the primary address of our STUN server with a change port and change address attributes. These binding request attributes solicit the server to send the response from the alternate IP address and port. If the client

receives the response, then the filtering behavior of the NAT is Endpoint-Independent. If not, we send a third binding request to the primary address with only change port. If the client receives a response then the NAT is AddressDependent Filtering, if not it is Port- Dependent Filtering.

4. **Unsolicited inbound SYN packets.** We perform two tests:
   - Send a SYN packet from our server, use an exponential back off time and verify we receive an ICMP Port Unreachable error for the original SYN.
   - Send a SYN packet from an out server, send a SYN packet from in for the connection and then verify that the SYN has been dropped. This test can be done only if the NAT has a port preservation behavior. Then it would assign the same port from the client to the NAT and the same port from the NAT to the server. We first verify if the NAT perform port preservation and then we send the SYN to the assigned port to that source port.

5. **Source port selection.** We send a packet using a given source port and then we compare the port number with the external bound port. If they match the NAT is performing port preservation. Moreover, if they are even or odds, the NAT implement port parity. If they do not match, it is possible that the NAT does not implement port preservation, but it is also possible that the specific port used as source port was already in use in another mapping (from another internal host) for this reason we will repeat the test several times. we also provide the percentage of port preserved. We verify if the port range is maintained by setting a source port in a given range and then check the external port.

6. **Supporting hairpinning.** In order to check it we send two packets: the first one to discover our mapped address using STUN protocol and the second one from a different source port and towards the discovered mapped address. If we receive the response it means the NAT supports hairpinning.

7. **Behaviour with respect to the ICMP Destination Unreachable.** We establish a TCP connection, then we send a ICMP Destination Unreachable message. We verify that the packet is forwarded and we check the payload to verify that the IP address and the port are translated in both directions (i.e., from the client to the server and from the server to the client).

8. **Mapping and ICMP packets.** We established a mapping and then we send an ICMP destination unreachable, then we check the mapping is still ok.

9. **TCP Connectivity with the STUN Server.** This test can be performed together with the other test. We simply send a STUN Binding Request to a TCP STUN server and verify the response.

10. **Packet modification.** To test if the NAT manipulates packets. We first run Revelio in order to understand the location of the CGN. Then we run tracebox forging different packets (adding TCP options or fixed sequence number) to understand which modifications the CGN implements.

UDP Requirements.

1. **UDP mapping.**

2. **Source port selection.**

3. **Mapping lifetime over 2 minutes.** We send a first binding request to our server and after 2 minutes we send another binding request using a different port but specifying that the response should be sent to the previous binding. If the answer is not filtered, it means that the previous mapping has been preserved.

4. **Outbound refresh behavior.** This test is similar to the previous one, except that an additional outgoing packet is sent one minute after the initial binding request and the second binding request is sent after 3 minutes.

5. **UDP filtering.**

6. **Supporting hairpinning.**

7. **Deterministic behavior.** We check the mapping and filtering behavior several times.

8. **Unreachable ICMP packets.** A device on the inside sends a packet to an external address that causes an ICMP Destination Unreachable packet to be returned. The test records whether this packet makes it back through the NAT correctly.

9. **Mapping and ICMP packets.** The methodology is the same than before unless that we are going to check the mapping after the ICMP unreachable packet is received.

10. **Supporting the "do not fragment" flag.** Our application sends a packet with the "Do not Fragment flag" set and waits for the reception of the correspondent ICMP message.

11. **Out-of-order UDP fragments.** In order to check this behavior, we send from our server disordered fragments and check their reception in the application.

12.

ICMP Requirements.

1. **ICMP query from internal.** The test verifies if the NAT supports simple ICMP Reply/Request message exchange by sending a query and waiting for the answer.
2. **ICMP query session less over 60 sec.** In order to check this, we send a first binding request to our server and after 2 minutes we send another binding request using a different port but specifying that the response should be sent to the previous binding. If the answer is not filtered, it means that the previous mapping has been preserved.
3. **Handling ICMP error packet.** We send an ICMP packet with wrong checksum and verify that the NAT drop the packet. a) send a ICMP packet with incorrect IP checksum and verify that the NAT drops the packet. c) check if a ICMP error message is sent.
4. **Refreshing mapping ICMP error packet.**We send an ICMP error packet from an external server with no mapping and verify that the NAT drops the packet. Then create a mapping and send a ICMP error packet check that the error type and code are unchanged and check that the destination IP address is the same as the source IP address of the embedded packet after translation.
5. **Error packet hairpinning.** We check this by sending two packets, the first one to discover our mapped address using STUN protocol and the second one is an ICMP Echo Error message sent to the mapped address from a different port. The NATwatcher verifies the reception of the error message.
6. **Support of ICMP destination unreachable.** We send an UDP packet to our server and the server replies with Destination Unreachable behavior. We then verify is the ICMP error is received by the CGNwatcher client.
7. **ICMP time exceeded.** Same as the previous one but for ICMP Time Exceeded messages.
8. **Timestamp and Timestamp Reply Messages.** We send a timestamp server function and check if the NAT returns a timestamp reply.
9. **Source route options.** We send a packet with source route option from outside and see if the packet it's forwarded correctly. If the message is not forwarded we verify that outbound it contains the route ip option.

10. **Address Mask Request/Reply Message.** We send a ICMP Address mask request and verify that the NAT answer with a ICMP Address mask reply.

11. **Parameter Problem Message.** We send a ICMP parameter problem message and verify that the NAT answer with a ICMP parameter problem message reply.

12. **Non-QueryError ICMP Messages.** We send a Non-QueryError ICMP Messages and see if the NAT handles or not those packets.

13.

CGNs Requirements.

1. **Limiting the number of external ports or identifiers for ICMP.** A CGN MUST support limiting the number of external ports

2. **Limiting the amount of state memory allocated per mapping and per subscriber.**

3. **Port reallocation.** Once an external port is deallocated, it SHOULD NOT be reallocated to a new mapping until at least 120 seconds have passed.

4. **Tracking the connection.** We can verify if the CGNAT is tracking the connection.

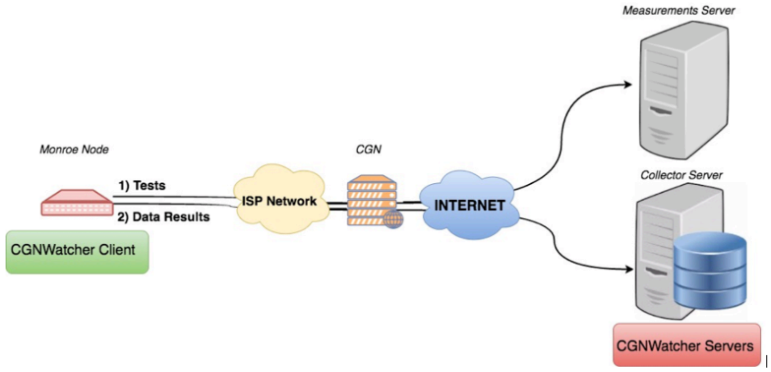
5. **Implementation of Port Control Protocol.**

6. **Unable to create mapping.** The CGN:

   - MUST drop the original packet.
   - SHOULD send an ICMP Destination Unreachable message with code 1 (Host Unreachable) to the sender;
   - SHOULD send a notification (e.g., SNMP trap) towards a management system (if configured to do so); and
   - MUST NOT delete existing mappings in order to "make room" for the new one. (This only applies to normal CGN behavior, not to manual operator intervention.).

7. **A CGN's port allocation scheme** SHOULD make it hard for attackers to guess port numbers.

Figure 9.1: CGNWatcher Operational Setup.



9.2 *Design, Development and Implementation of CGNwatcher*

Figure 9.1 summarizes the operational setup of CGNWatcher.

In a nutshell, we create the CGNWatcher Docker container necessary to run the experiment in the MONROE node. The Docker container executes the CGNWatcher application. The CGNWatcher application automatically executes the tests to characterize the CGN behaviour, by sending different combinations of packets to our Measurement server deployed in the public Internet and processes the response packets sent from the server back to the MONROE node. The server does not produce any experiment results. Once all the tests are completed the application create an output results file and sends the compiled measurement results to a Collector server where they are stored. All tests are started by the CGNWatcher client on the MONROE node. Most experiments use stream sockets on the server side, some of them also use python-scapy, meaning raw sockets. The goal of CGNWatcher is to enhance the MONROE platform with the ability to measure CGN properties that cannot be measured in other crowdsourcing measurement platforms. For example, MONROE offers the possibility to repeat the tests over time and to decide when to run the tests (day or night).

In this Section we describe the methodology we used for testing each requirements.

The goal of the behavioural tests is to determine the real implications for the end users. We implement the tests to have a short execution time, in order to make the MONROE usage efficient.

The behavioural tests are the following.

1. **Port exhaustion test.** Behind a CGN there are multiple devices that have to share the port space among each other. This test estimates the maximum number of mappings that can exist for one device behind the CGN by opening any given number of connections and waiting for periodical responses. We create a mapping every 0.1 second until reach the given number of connections. We then send a keep alive message to maintain the mapping. Therefore, we observe if new connections are rejected or old mappings are eliminated in favour of new ones. We also observe if the mapping is eliminated silently or if any ICMP error message or RST is sent. After experimenting the test on MONROE development nodes and observing that the MNOs do not block the user, we consider the same test for different traffic patterns to verify if this influences the CGN on removing existing mapping in favour of new ones. For example: for a few selected connections we send a small amount of data and for the remaining connections we send data more frequently or bigger packets. We then observe the behaviour of the CGN.

2. **Mapping lifetime on different ports tests.** The goal of this test is to determine the configured mapping lifetime. NATs are required to maintain UDP mappings during no less than 2 minutes and TCP mapping during no less than 2 hours and 4 minutes to follow the IETF requirements. In order to check this, we create a mapping from the MONROE node. After receiving the initial packet, the server waits for a fixed amount of time before responding with another packet to test the liveliness of the mapping. When we test TCP we consider 120 minutes, incrementing the time to send the keep alive every 10 seconds. When we test UDP we consider 10 minutes, incrementing the time to send the keep alive every 10 seconds. According to the specifications of the CGNs, the maintainer can set different mapping lifetime depending on the port used. For this reason, we check the mapping lifetime in different ports such as 53 (DNS port), 80 (HTTP port), 443 (HTTPS) and 5228-5230 (Android application notification port).

3. **Port allocation strategy tests.** This test checks if the CGN leaves the external port unchanged when creating a mapping. We send a packet using a particular local source port and then we compare the port number with the external bound port. If they match the CGN is performing port preservation. If they do not match, it is possible that the CGN does not implement port preservation, but

it is also possible that the specific port used as source port was already in use in another mapping (from another internal host). Because of this limitation, this test only provides a lower bound to the number of CGNs that implement port preservation. It was possible to increase the accuracy of this test by repeating the test several times, thanks to the repeatability property that MONROE offers. It is also possible that the CGN implements a 'common-not-random' behaviour. For this reason, we have a deeper look at allocation ports. We allocate a block of port first randomly and then sequentially.

4. **IP allocation strategy tests.** The test checks if the assigned public IP address changes over time and the average duration of the IP address for each MNO. The test is as follows: we first send a STUN binding request. We then save the IP address returned in the STUN response received. The STUN protocol operates as follows: the client located behind the CGN sends a binding request message to a STUN server. The STUN server replies with a response message containing the IP address and port of the client in its payload, as observed from the server. We implement CGN-Watcher using python and the packet generator python-scapy. In particular, we design it to be as simple as possible in order to be used not only in MONROE, but also in other measurement platforms.

The output produced by the tests is in the JSON format which allows the file to be parsed easily. We built an evaluation script that analyses and summarizes the results.

9.3 CGNwatcher: Results and Dataset

9.3.1 CGNwatcher Measurements

We initially implement the tests on 5 operators: Orange ES, Yoigo ES, Vodafone IT, Telia NO and Telenor NO on dev nodes. Once verified that the tests are safe (the link was not broken during the test) we implement the tests on the rest of the MONROE nodes. The CGNWatcher container was created to be simple and scalable. It is possible to specify the parameters of the tests using python flags:

$cgnwatcher_client.py - iINTERFACE - sSERVER - oOUTPUT - port_exhaustionPORTS - max_timeoutMAX_TIMEOUT$

In order to shed some light about CGN while roaming we run the set of measurements, we first need to configure the nodes by activating and deploying the SIMs. For each MNO, we perform the measurements at the same time from all six countries and coordinate the configuration of the experimental setup. We repeat the measurements ten times towards each target for each SIM in the visited country and in the home country. The resulting dataset lists the set of IP hops along the data paths from each vantage point towards each measurement responder.

9.3.2 *Analysis of the Measurement Results*

The output produced by the tests is in the JSON format which allows the file to be parsed easily. We built an evaluation script that analyses and summarizes the results. In Table 1 we summarize the MNOs and report whether the user is behind a CGN or not. If the assigned local IP address is public, then the user is not behind a CGN. Results show that Telenor NO, Telenor SE and Vodafone IT are not behind a CGN. This is quite uncommon for an MNO that usually has a limited pool of IP addresses to assign.

Requirements Tests Results. Table9.1 summarizes the results from the requirements tests.

CGNWatcher test/MNO	Orange	Yoigo	Movistar	ICE	Telia NO	Telia SE	Three	TIM
Mapping behavior TCP	E.I.	A.P.D.	E.I.	E.I.	A.P.D.	A.P.D.	A.P.D.	E.I.
Mapping behavior UDP	E.I.	A.P.D.	E.I.	E.I.	A.P.D.	A.P.D.	A.P.D.	E.I.
Filtering behavior TCP	A.P.D.	A.P.D.	A.P.D.	A.P.D.	A.P.D.	A.P.D.	A.P.D.	A.P.D.
Filtering behavior UDP	A.P.D.	A.P.D.	A.P.D.	A.P.D.	A.P.D.	A.P.D.	A.P.D.	A.P.D.
Simultaneous Open Connection	NO	NO	NO	NO	NO	NO	NO	NO
Unsolicited inbound SYN packets	YES	YES	YES	YES	YES	YES	YES	YES
Hairpinning TCP	NO	NO	NO	NO	NO	NO	NO	NO
Hairpinning UDP	NO	NO	NO	NO	NO	NO	NO	NO
Behaviour with respect to the ICMP Destination Unreachable TCP	YES	YES	YES	YES	YES	YES	YES	YES
Behaviour with respect to the ICMP Destination Unreachable UDP	YES	YES	YES	YES	YES	YES	YES	YES
Mapping and ICMP packets TCP	YES	YES	YES	YES	YES	YES	YES	YES
Mapping and ICMP packets UDP	YES	YES	YES	YES	YES	YES	YES	YES
Packet modifications TCP	NO	NO	NO	NO	NO	NO	NO	NO
Packet modifications UDP	NO	NO	NO	NO	NO	NO	NO	NO
Deterministic behavior TCP	YES	YES	YES	YES	YES	YES	YES	YES
Deterministic behavior UDP	YES	YES	YES	YES	YES	YES	YES	YES
Supporting the "do not fragment flag"	YES	YES	YES	YES	YES	YES	YES	YES
ICMP query from internal	YES	YES	YES	YES	YES	YES	YES	YES
Handling ICMP error packet	YES	YES	YES	YES	YES	YES	YES	YES
Refreshing mapping ICMP error packet	YES	YES	YES	YES	YES	YES	YES	YES
ICMP time exceeded	YES	YES	YES	YES	YES	YES	YES	YES
Timestamp and Timestamp Reply Messages	YES	YES	YES	YES	YES	YES	YES	YES
Address Mask Request/Reply Message	YES	YES	YES	YES	YES	YES	YES	YES
Parameter Problem Message	YES	YES	YES	YES	YES	YES	YES	YES
Non-QueryError ICMP Messages	YES	YES	YES	YES	YES	YES	YES	YES

Table 9.1: Results from the requirements tests.

All tests were performed 97 times during the day and during the night. We run a first campaign during December 2017 and a second campaign during the month of June 2018. Results from requirements tests are similar among MNOs. Most MNOs use a restrictive CGN configuration like an Address and Port-Dependent mapping and filter (Yoigo, Telia NO, Telia SE and 3). The rest uses a more open configuration: Endpoint-Independent mapping and Address and Port-Dependent filter. No CGNs allows hairpinning or implement packets modification. All the MNOs have a deterministic behaviour, the CGN maintains the same mapping or filtering during time. All kinds of ICMP packets are supported.

Behavioural Tests Results.

1. **Port exhaustion results.**

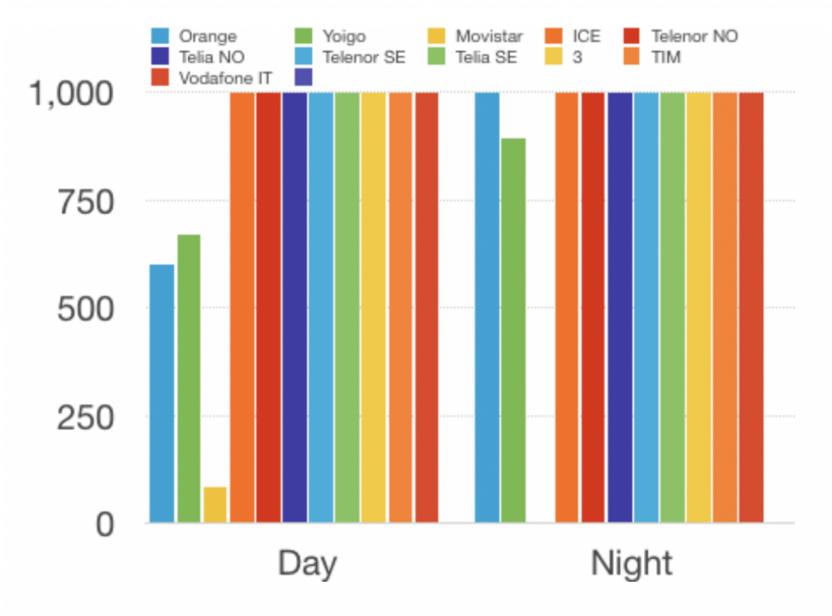


Figure 9.2: Port exhaustion results.

We run the test both during day and night. Figure9.2 shows that Orange and Yoigo are allocating fewer ports during the day.

We then evaluate why the maximum number of ports that we can use differs on the same MNO. According to the specifications of the CGNs, the maintainer may choose to reserve a subset of the total port quota, thus freeing the remainder of the ports to be used by another user. For example, one hundred ports can be immediately reserved while the remaining 900 ports are free to be used by other clients.

Next, we try to understand what happens once the ports limit is reached. Orange and Movistar are discarding older connections when the maximum number of mapping is reached, in order to make room for new connections initiated.Yoigo refuses new connections and keeps the existing ones. Moreover, Orange and Movistar discard old connections silently (i.e. no ICMP, RST packets detected), while Yoigo sends an ICMP error message. When different traffic patterns are tested the situation is unchanged. This means that not specific strategies for different traffic pattern are used by the CGN.

2. **Mapping lifetime on different ports results.** Figure9.3 and Figure9.4 show the mapping lifetime for each MNO for the TCP and UDP tests respectively. Results show that TIM has a short mapping lifetime. As the RFC specifies the UDP mapping lifetime must not expire in less than two minutes. All MNOs follow this requirement.

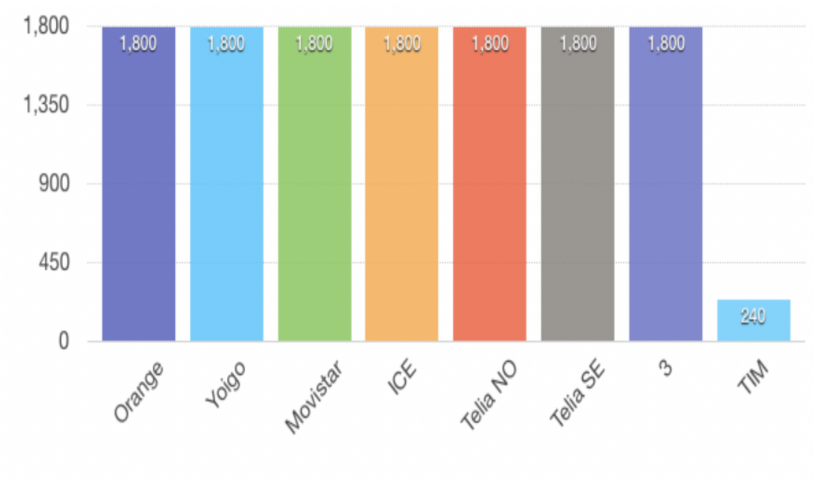


Figure 9.3: TCP mapping lifetime per MNO.

3. **Port allocation strategy tests results.** We observe that the CGNs allocate both registered and ephemeral ports (1024 - 65535). No MNOs, apart from the ones that are not using a CGN, implement port preservation. Yoigo is the only MNO that implements a non-random port allocation strategy. Figure9.5 shows the internal ports assigned by us and the corresponding mapped ports assigned by the CGN. The ports are assigned sequentially. Results do not change with time. The same ports are mapped when we assign the same internal port after 3 hours.

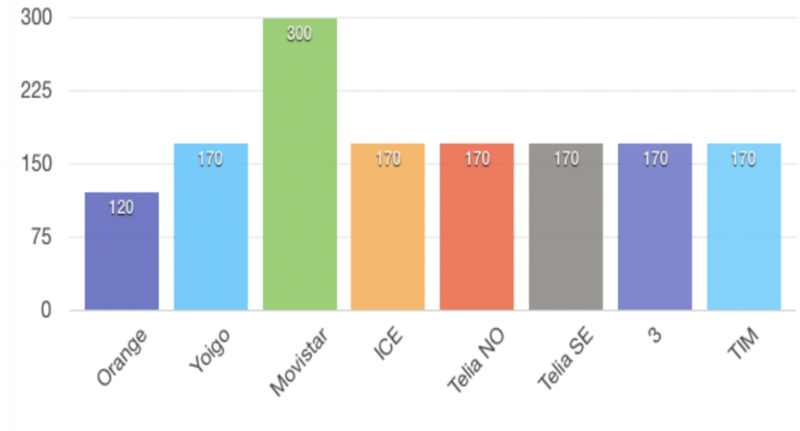


Figure 9.4: UDP mapping lifetime per MNO.

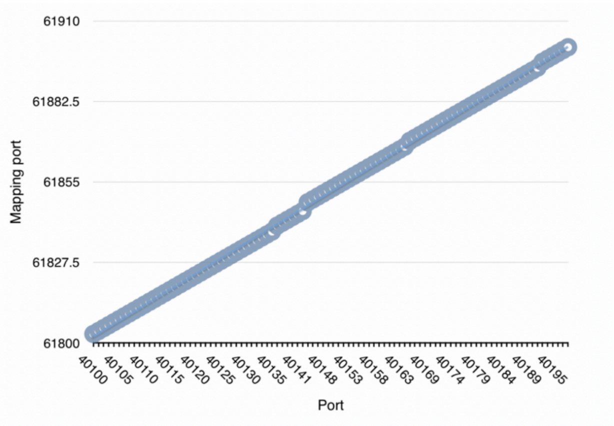


Figure 9.5: Yoigo port mapping.

9.4 Lessons Learned

We created a measurement test suite for the analysis of CGN in cellular network. While the experiments were designed to run in the MONROE nodes, the test suite can also be used on other measurement platforms. Moreover, our container is easily extendable. During the development of the test suite we gained experience with the MONROE testbed. The results show different configuration between MNOs. In general, the mapping timeouts are lower than recommended by the RFCs. Our results showed similar behaviour for TCP and UDP.

The integration of the test suite into the MONROE testbed could be improved further. Our tests could be executed in parallel for all available same MNOs and therefore yield more results on the strategy for the same MNO in the same Country.

Part III Implication of Roaming in Europe

10 Implication of Roaming in Europe

Roaming allows mobile users to use their voice and data services when visiting a network other than their home network. Roaming can be national or international. International roaming allows mobile users to use their voice and data services when they are abroad. The former takes place within a users home country, whereas the latter allows users to access their services also when they are abroad. While national roaming is virtually transparent to users, international roaming can be associated with high costs.

The EC, in an effort to create a single digital market across the EU, has recently (as of June 2017) introduced a set of regulatory decisions [1] as part of the "Roam like Home" initiative. This initiative abolishes charges for users when they use voice and data services while roaming in EU.

It is designed to prevent unexpected charges due to extra mediation and billing costs when roaming services are active. In this setting, MNOs are expected to deliver services with QoS properties similar to the ones a user experiences when at home.

In this work, we perform an extensive large-scale measurement study to understand the roaming ecosystem in Europe after the "Roam like Home" initiative. More specifically, we investigate:

- Which technical solutions are actually being deployed and used today?
- What are the implications of roaming on the service experienced by the roaming user?

Figure 10.1: Roam like Home.

10.1 Roaming background

To support roaming, MNO commonly connect with each other through an IPX network. An IPX. [2], [3] can be described as a hub that interconnects MNO over a private IP backbone network and is possibly run by a third party IPX provider. An IPX provider has

[1] European Commission: New Rules on Roaming Charges and Open Internet. https://ec.europa.eu/digital-single-market/en/news/new-rules-roaming-charges-and-open-internet. [Online; accessed 06-March-2018]

[2] GSM Association: IPX White Paper. https://www.gsma.com/iot/wp-content/uploads/2012/03/ipxwp12.pdf, b. [Online; accessed 06-March-2018]

[3] GSM Association: Guidelines for IPX Provider Networks. https://www.gsma.com/newsroom/wp-content/uploads//IR.34-v13.0-1.pdf, a. [Online; accessed 06-March-2018]

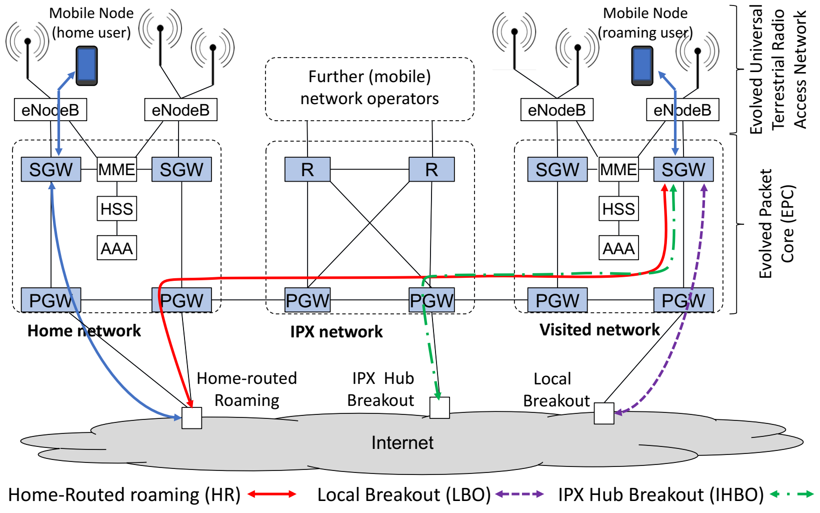


Figure 10.2: Internet access options for a mobile node at home (left) and when roaming (right).

connections to multiple network operators and thus enables each MNO to connect to other operators via a single point of contact.

The interconnections between MNOs are accompanied by roaming agreements that enable the operators to apply policies, control network access for roaming subscribers, and manage their roaming services. Figure 10.2 illustrates the three main schemes that can be employed for providing data roaming services and are further described below, namely, HR, LBO and IHBO.

When a mobile node is at home (left, see Figure 10.2), the *home user*'s traffic will take a short path inside the network to reach a suitable PGW to the Internet. The traffic of a *roaming user* (right, see Figure 10.2) is directed to an egress PGW whose location depends on the roaming architecture. In the case of HR, the mobile node receives the IP address from its home MNO and the *roaming user*'s traffic is first routed towards a PGW in the *home network* (red path).

When LBO [4], [5] is used, the IP address of the roaming user is provided by the visited network. The GTP tunnel is terminated at the PGW of the visited network and IP-based services can be accessed directly from there (purple path). This does not add latency and reduces network resource usage, but may restrict access to private services in the user's home network. Service control and charging also become more complex using LBO.

IHBO [6] provides an alternative to overcome the limitations of home-routed roaming and local breakout. Here, the IP address of the roaming user is provided by the IPX network. The GTP tunnel from the Serving Gateway (SGW) in the visited network terminates

[4] GSM Association: LTE and EPC Roaming Guidelines. https://www.gsma.com/newsroom/wp-content/uploads/IR.88-v15.0.pdf, a. [Online; accessed 06-March-2018]

[5] Huawei: LTE International Roaming Whitepaper. http://carrier.huawei.com/en/technical-topics/core-network/LTE-roaming-whitepaper, b. [Online; accessed 06-March-2018]

[6] Method and System For Hub Breakout Roaming. https://patents.google.com/patent/US20140169286/en. [Online; accessed 06-March-2018]

at a PGW in the IPX network (green path). There may be multiple PGWs so that latency and resource usage can be reduced by selecting one geographically close to the visited network. As the IPX network maintains a trusted relationship with the home network, it may assign an IP address recognized by the home network to the roaming user, thereby allowing the user access also to private services in the home network. IHBO can also simplify setup and management as a single GTP tunnel, terminated in the IPX network, can be used for roaming users from different home networks.

The topology can have a potential impact on the communication performance. For instance, when the node accesses services inside the *visited network*, the performance is likely to be worse in the HR case, because all packets travel twice between the visited and the home country; less so when the communication peer is in a third country and is minimal when accessing services in the home country. This may also have implications in the selection of Content Delivery Network (CDN) when roaming abroad, because the mobile user will access a server in the home network rather than one close to their location.

In this Chapter, we present the hardware platform we built for roaming measurements, and the manner in which we orchestrate it to collect our data.

10.2 MONROE-Roaming Platform

We design and build MONROE-Roaming, a dedicated platform for roaming measurements in Europe. MONROE-Roaming integrates several components that we depict in Figure 10.3.

The main blocks include measurement nodes distributed in six different EU countries, the backend system, several measurement servers and a scheduler, all of which we detail next. To build the MONROE-Roaming platform we adapted the open source software provided by MONROE. [7]

MONROE-Roaming nodes: Each MONROE-Roaming node is equipped with an APU board from PC Engines with two 3G/4G MC7455 LTE CAT6 miniPCI express modems. Because of the high cost of nodes and subscriptions, and the complexity of the coordination effort required (see subsection 10.2.2), we have set up a platform with a total of 12 MONROE-Roaming nodes dedicated for roaming measurements.

MONROE-Roaming backend: Upon completion of each measurement, MONROE-Roaming nodes transfer the measurement results to a central server for further analysis.

Measurement servers: We have deployed one measurement server

[7] Ö. Alay, A. Lutu, M. P. Quirós, V. Mancuso, T. Hirsch, K. Evensen, A. F. Hansen, S. Alfredsson, J. Karlsson, A. Brunstrom, A. S. Khatouni, M. Mellia, and M. A. Marsan. Experience: An Open Platform for Experimentation with Commercial Mobile Broadband Networks. MobiCom 2017, pages 70–78, 2017b. DOI: 10.1145/3117811.3117812

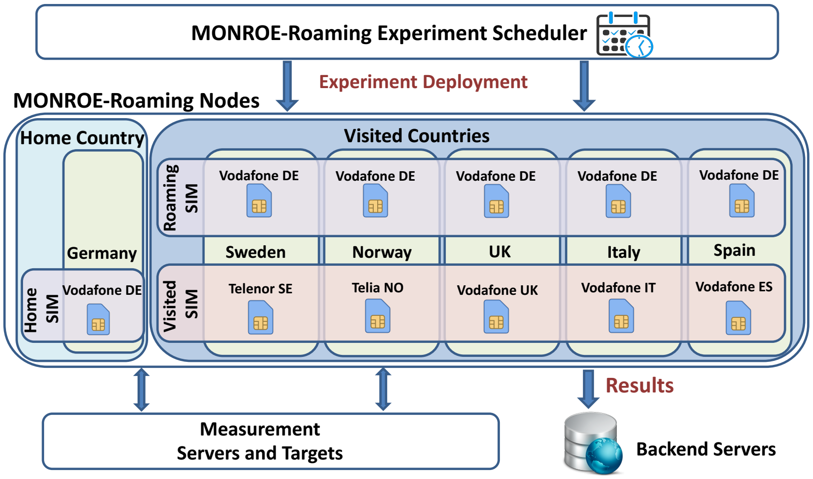
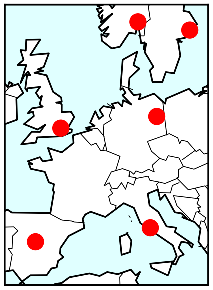


Figure 10.3: MONROE-Roaming platform and experimental setup. We exemplify our setup for Vodafone DE. We have five Vodafone DE SIMs in international roaming nodes and another SIM in the home country nodes. For each roaming Vodafone DE SIM, we insert the SIM corresponding to the local roaming partner for the MNO. For example, in Sweden we use the Telenor SE SIM which corresponds to the network on which the Vodafone DE SIM is camping.

in each country as measurement responders and also to capture traffic traces.

MONROE-Roaming scheduler: The scheduler allows the user to query for resources, select nodes and launch different tests in the platform simultaneously. We used the open source MONROE scheduler as a basis for the MONROE-Roaming scheduler. Each test is designed and implemented in a Docker container. [8]

10.2.1 *Experimental Setup*



	MNO		
NO	Telia NO	Telenor NO	
SE	Telia SE	Telenor SE	3 SE
UK	Vodafone UK	EE	
DE	Vodafone DE	T-Mobile	O2
ES	Vodafone ES	Movistar	Orange
IT	Vodafone IT	TIM	3 IT

Figure 10.4: The distribution (left) of the MONROE-Roaming nodes in six countries and (right) SIMs for 16 MNO we measure across Europe. Each country deploys two MONROE-Roaming nodes and one measurement server.

To understand the roaming ecosystem in Europe, we focus on di-

[8] D. Merkel. Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux Journal*, 2014 (239), Mar. 2014. ISSN 1075-3583

versity of the MNO. In other words, we aim to cover a large number of SIM rather than running measurements from a large number of vantage points. To this end, we deployed two MONROE-Roaming nodes in each of the six European countries to measure a total of 16 MNO that operate their own network, as illustrated in Figure 10.4.

For each MNO, we bought six SIMs that support roaming in Europe and we distributed one SIM in each of the countries we cover. For example, in Germany, we bought six Vodafone DE SIMs that support roaming. We kept one Vodafone DE SIM as the *home SIM* in the home country (i.e., Germany). Then, we distributed five *roaming SIMs* from Vodafone DE to the other five countries (i.e, Sweden, Norway, UK, Italy and Spain). Each roaming SIM connects to (or *camps on*) a local roaming partner (or visited network) native to the visited country. For example, Vodafone DE in Germany is a roaming partner of Telenor SE in Sweden. Therefore, Telenor SE serves Vodafone DE's customers roaming in Sweden by allowing Vodafone DE users to camp on Telenor SE's network. For each roaming SIM, we identify the corresponding visited network (e.g., Telenor SE in Sweden for Vodafone DE) and, when available, activate the corresponding native SIM from the visited network (which we hereinafter denote by *visited SIM*). We illustrate this configuration in the experimental setup in Figure 10.3. We also describe the terminology in Table 10.1.

Table 10.1: Terminology.

Home network	The network to which a mobile user subscribes.
Home SIM	The mobile user SIM while in the home country and connected to the home network.
Visited Network	The network to which a user connects while roaming internationally in the visited country.
Visited SIM	A user subscribed to visited network in the visited country.
Roaming User	A user subscribed to a home network in *country A* and who is roaming in a foreign *country B*.

10.2.2 *Measurement Coordination*

Each MNO-specific measurement campaign involves 11 SIM and 6 nodes: (i) one node with the home SIM and (ii) five nodes with both the roaming SIM and the corresponding visited SIM, as illustrated in Figure 10.3. This enables us to capture performance metrics for the roaming SIM, but also to compare those with the local performance of the home network and the visited network (when possible).

Before running the set of measurements (see section 10.3, section 10.4 and section 10.5), we first need to configure the nodes by activating and deploying the SIM. For each MNO, we perform the measurements at the same time from all six countries and coordinate the configuration of the experimental setup in two steps:

Home and Roaming User Activation: To measure a MNO, we first insert the SIM into the first SIM slot in each node in all six deployment locations. For the SIM active in its home country, this step triggers the home user activation (by inserting the SIM in the measurement node). For the rest of the nodes, this step triggers the roaming user activation.

Visited User Activation: Once we complete the home and roaming user activation, we check which visited network the roaming SIM uses in each of the five visited countries. Then, we insert the SIM of each partner MNO (when available) into the second slot of each corresponding node.

Using the MONROE-Roaming scheduler, we orchestrate the execution of the measurements so that they run in parallel on all nodes. The measurement coordination effort was a significant part of the process. In each country, at least one person was dedicated to carry out the physical experimental setup configuration for each MNO in a timely manner. Given that we deploy two nodes per country, we could measure two MNO and (maximum) 22 SIM in parallel. We coordinated the SIM changes over email. Furthermore, before the change of the next pair of SIM, we double-checked the measurement results we had collected to ensure correctness and completeness of the dataset. Each round lasted one week, over a total period of more than four months of experiments.

10.3 Roaming Setup and Performance

10.3.1 Methodology

We run a series of measurements that enable us to identify the roaming setup, infer the network configuration for the 16 MNO that we measure and quantify the end-user performance for the roaming configurations which we detect. We run `traceroute` for path discovery, `dig` for DNS lookups and `curl` for testing data transfers with popular URLs. We complement this analysis with metadata (e.g., radio access technology, signal strength parameters) collected from each node.

For each MNO, we measure in parallel the roaming user, the home user and the visited user (see Table 10.1 for terminology) through the MONROE-Roaming scheduler. In this way, we are able to capture potential performance penalties that might result, for example, from roaming internationally under a home-routed configuration. We performed measurements using both 3G and 4G networks to evaluate the impact of potentially different configurations for the two radio access technologies.

Next, we describe each measurement test and its resulting dataset in more details.

traceroute. We run periodic `traceroute` measurements against all the servers we deploy in each country as measurement responders. We repeat the measurements ten times towards each target. The resulting dataset lists the set of IP hops along the data paths from each vantage point towards each measurement responder. Additionally, we collect the public mapped IP address for each vantage point (i.e., the IP endpoint associated with the mobile client as seen from the public Internet).

dig. We run the `dig` utility for DNS lookups against a list of 180 target FQDN mapped to advertisement services. We use the independent filter lists from `https://filterlists.com` to build the list of targets. We focus on ad services because this type of third party services inflate significantly performance metrics of web services (e.g., page load time), as well as impact the web experience of mobile users. [9] Thus, it is important to captute (and potentially eliminate) any additional delay penalty that might impact how fast a roaming user receives this type of content. Each experiment uses the default DNS server for the tested MNO and queries for the A record associated to each of the target FQDN. We store the entire output of each `dig` query, including the query time, the DNS server used and the A record retrieved. We repeat the `dig` queries 2 times for each FQDN from each vantage point, for a total of more than 2,000 queries per round.

curl. We run `curl` towards a set of 10 target popular webpages[10] over HTTP1.1/TLS. We repeat the measurements towards each URL at least 10 times (increasing the sample size if the SIM data quota allows it). We store various metrics, including the download speed, the size of the download, the total time of the test, the time to first byte, the name lookup time (query time) and the handshake time.

metadata. We collect contextual information from the nodes, including the visited network MCC / MNC for each roaming SIM and the radio technology. This allows us to verify which visited network

[9] U. Goel, M. Steiner, M. P. Wittie, M. Flack, and S. Ludin. Measuring What is Not Ours: A Tale of 3^{rd} Party Performance. In M. A. Kaafar, S. Uhlig, and J. Amann, editors, *Passive and Active Measurement*, pages 142–155. Springer International Publishing, 2017

[10] We target the following web pages: `www.httpvshttps.com`, `facebook.com/telia/`, `en.wikipedia.org/wiki/Timeline_of_the_far_future`, `linkedin.com/company/facebook`, `www.yahoo.com/movies`, `instagram.com/leomessi/`, `google.com/search?q=iPhone+7`, `youtube.com/watch?v=xGJ5a7uIZ1g`, `ebay.com/globaldeals`, `nytimes.com`, `theguardian.com.uk/lifeandstyle`.

each roaming SIM uses as well as to identify and separate the collected data by radio technology.

10.3.2 Roaming configuration

Our initial goal is to determine the roaming setup for each MNO (i.e., whether it used LBO, HR or IHBO). For this, we determine the MNO that allocates the public IP address of the roaming SIM. Our results show that *HR was used by all 16 MNOs from all the different roaming locations we capture*. We further corroborate this result by retrieving the first hop replying with a public IP address along the data path from a roaming SIM to each server and identifying the MNO that owns it. We find that the first hop with a public IP address along the path lies in the original home network of each roaming SIM, which is consistent with HR.

Next, we evaluate the following performance metrics for each roaming SIM, home SIM and visited SIM: (i) the number of visited networks we observe for the roaming SIM, (ii) the number of hops from vantage point to target measurement server, (iii) the number of home network PGWs that the roaming SIMs reach in comparison with the home network SIMs.

Visited network selection: The metadata we collect during the measurement campaign for each MNO enables us to verify the visited network that each roaming user camps on in the visited country. In general, we note stability both in 4G roaming and 3G roaming in the selection of the visited network (Table 10.2) in the five roaming locations. We also observe some differences between MNOs. For example, for Telekom DE, the 4G visited network chosen by each roaming SIM never changed during the measurement campaign, even when we forced the radio technology handover. This is consistent for all the five roaming locations. For O2 DE, on the other hand, the default 4G visited network did change over time for the SIMs roaming in Italy (3 visited networks), Norway (3 visited networks), and Sweden (2 visited networks). However, it should be noted that the length of the measurement period varies for each MNO, as it is impacted by multiple external factors (e.g., at times some of our measurement responders were affected by power outages or some SIM cards were not connecting to the 4G network due to poor coverage). This may explain or influence part of the differences observed between the MNOs.

Traceroutes, number of hops: We analyze our collected traceroute results from the roaming SIMs and compare with the traceroute results we collect from the corresponding home SIM towards the same target server. For all MNOs we find that *the number of hops is the*

MNO	# of visited networks		# IP addr.	# Tests	First hop breakdown(%)
	3G	4G			
O2 DE	9	9	20	657	1; 1; 1; 2; 2; 2; 2; 2; 3; 3; 3; 4; 4; 5; 6; 6; 9; 9; 11; 24
Telekom DE	5	5	4	1424	13; 19; 25; 43
Voda DE	5	6	2	1511	46; 54
Movistar ES	6	6	8	282	4; 5; 5; 7; 8; 21; 22; 28
Orange ES	7	7	3	900	6; 43; 51
Voda ES	5	5	1	1943	100
TIM IT	6	6	4	497	1; 1; 46; 52
Voda IT	5	5	4	759	19; 19; 23; 39
Telenor NO	5	5	3	398	8; 30; 62
Telia NO	5	5	4	379	7; 16; 38; 39
3 SE	7	6	2	828	44; 56
Telenor SE	5	5	2	1362	32; 68
Telia SE	5	5	4	379	7; 16; 38; 39
EE UK	5	5	9	1038	3; 4; 4; 5; 8; 13; 17; 19; 27
Voda UK	5	5	1	503	100

Table 10.2: Distribution of the first IP interface and visited network per MNO. We report the number of networks each roaming user camps on in the visited country (# of visited networks), the number of unique first IP addresses (# IP addr.), the total number of traceroutes we ran for the corresponding SIM (# tests) and the distribution for each first IP address we find (First hop breakdown(%)).

same. [11] This is consistent with the HR configuration (Figure 10.2), where the GTP tunnel is defined between the SGW of the visited network and the PGW of the home network.

Traceroutes, infrastructure: By learning the IP addresses of the infrastructure elements along the data path, we are able to infer aspects of the infrastructure deployment strategy of each MNO. In particular, by checking the IP address of the first hop in the path (Table 10.2), we find that MNOs have different strategies in terms of their deployments. We note that the first hops have an even distribution on their assignation to mobile users, showing that the MNOs have a similar approach for load balancing in their network. For example, for O2 DE we find 20 different first hops, suggesting that there might be

[11] Traceroute for 3 IT did not work in any country to any server.

a large number of PGWs deployed in the LTE infrastructure, while for Vodafone UK we see that the same first hop appears on the data path, suggesting that the GTP tunnels of all our roaming users is terminated at a single PGW. We also note that although for the majority of MNOs, these hops are configured with private address space, three operators (Telekom DE, Telenor NO and Telenor SE) use public address space for their infrastructure. The last column in Table 10.2 details the breakdown of measurements among the number of different first hop IP addresses found. In some cases, a clear bias exists.

Finally, we verify that the set of first hops for roaming SIMs is the same as the set we observe from the home SIMs. This suggests that the roaming SIMs do not receive any differential treatment in terms of allocation to the PGWs. This is consistent for all MNOs we measure. Furthermore, when checking the 3G data paths, we find that the set of IP addresses we see in 3G is a subset of the set of IP addresses we see in 4G, suggesting that the two functions are co-located in the same PGW. [12] We also check the time when the first IP address was used. We discover that all the PGWs are active in the same time. Multiple first IP addresses can be used at different time. We further contacted 3 MNOs and the information they provided about their network confirms our findings.

[12] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, C. Ziemlicki, and Z. Smoreda. Not All Apps Are Created Equal: Analysis of Spatiotemporal Heterogeneity in Nationwide Mobile Service Usage. CoNEXT 2017, 2017

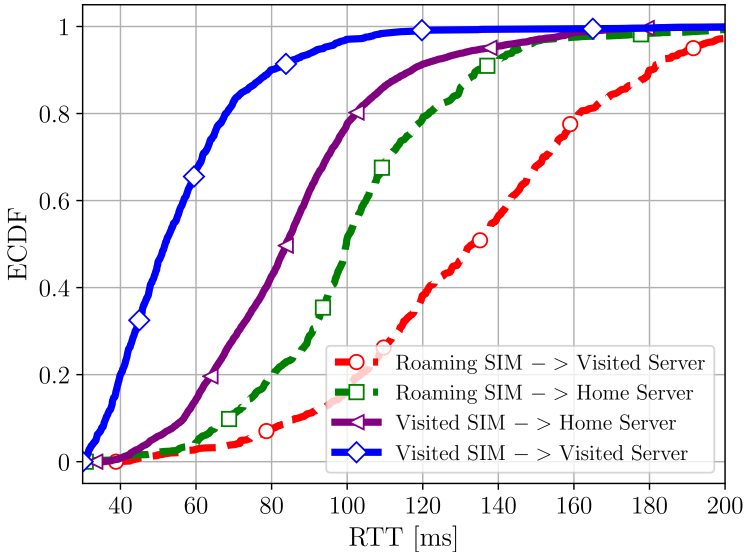


Figure 10.5: ECDF of the RTT from mobile nodes to target servers.

10.3.3 Home-Routed Roaming: Implications

Delay implications: The HR data implies that the roaming user's exit point to the Internet is always in the original home network (Figure 10.2). Thus, the data that the roaming user consumes always flow through the home network. Depending on the location of the server, this translates to a potential delay penalty. Figure 10.5 shows the ECDF of the RTT we measured between the roaming SIMs and the target servers located in the visited or home networks (red and green curves, respectively). To compare the HR with the LBO configuration, we also include the RTT measurements between the visited SIMs against the same targets in the visited or home networks (blue and purple curves, respectively). The RTTs experienced by the visited SIMs serve as estimates of the best RTTs that one could expect with a LBO configuration, since LBO relies on access to local infrastructure with no need for tunnelling back to the home network. We note that the largest delay penalty occurs when the roaming user tries to access a server located in the visited country. This is because the packets must go back and forth from the home network. Surprisingly, we note that the HR configuration also impacts the case when the roaming user accesses a target server located in the home network. That is, the GTP tunnel is slower than the native Internet path. In this case, the median value of the delay penalty considering all the MNOs is approximately 17ms. This varies across MNOs and in some cases we observe very low penalties (e.g., just 0.2ms for O2 Germany).

We investigate this performance impact further and calculate the estimated delay penalty between LBO and HR when the target is in the visited network. In more detail, we compute the delay penalty as the difference between the median delay to reach a given server when roaming, and the median delay to reach the same server from home. Figure 10.6 exemplifies these median values for Vodafone Germany. We note that, in general, the delay penalty varies widely with the geographical location of the roaming users and the target servers. For example, when a German SIM roams in Spain, the difference in terms of RTT is higher if the server is in the visited country (i.e., Spain) (red curve in Figure 10.5). If the German SIM roams in Spain or Italy and the target server is in Norway or Sweden the delay penalty of the roaming is smaller, since to go to Norway or Sweden the data path would anyway likely pass through Germany (and this is similar to the delay one would have because of the HR configuration).

We then evaluate the RTT difference between the roaming SIM and the visited SIM towards the same target and we group them per MNO. Figure 10.7 shows the median value of the delay penalty of an MNO (on the x axis of the tile plot) while roaming against each

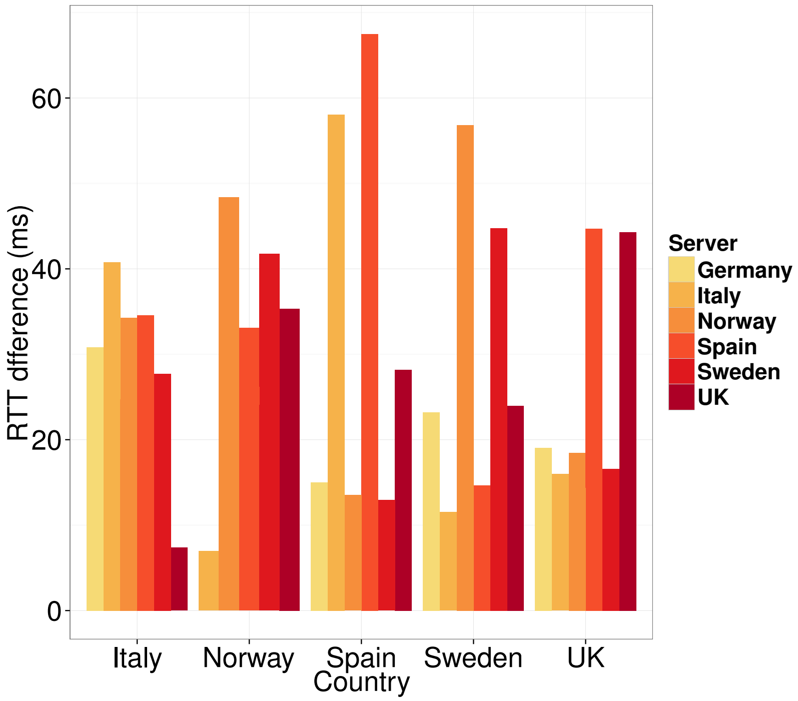


Figure 10.6: RTT difference from the visited country to all servers for Vodafone DE.

of the six different servers (on the y axis of the tile plot, marked by country). We note that the delay penalty varies as a function of the location of the home country. For example, German SIMs experience a lower delay penalty, which is potentially due to them being in an advantageous position in the center of Europe.

DNS implications: The results of the `dig` measurements show that the DNS server offered to a roaming user is the same as the one offered when at home. This is again consistent with the use of HR. We verify whether this translates into an inflated query time for the roaming user. Figure 10.8 presents the distribution of DNS query times for all the SIMs of TIM IT. We note that for the home user the query time is significantly lower in average than for the other five roaming users. This is consistent for all the 16 MNOs we measured. This further translates into implications in terms of CDN replica selection: the roaming user would be likely redirected to CDN content at its home network, and will not be able to access the same content from a local cache (which would in any case result in facing a higher delay due to the home routing policy).

HTTP performance implications: Similar to the delay and DNS implications, international roaming affects HTTP and HTTPS performance. We quantify this penalty by considering the handshake time

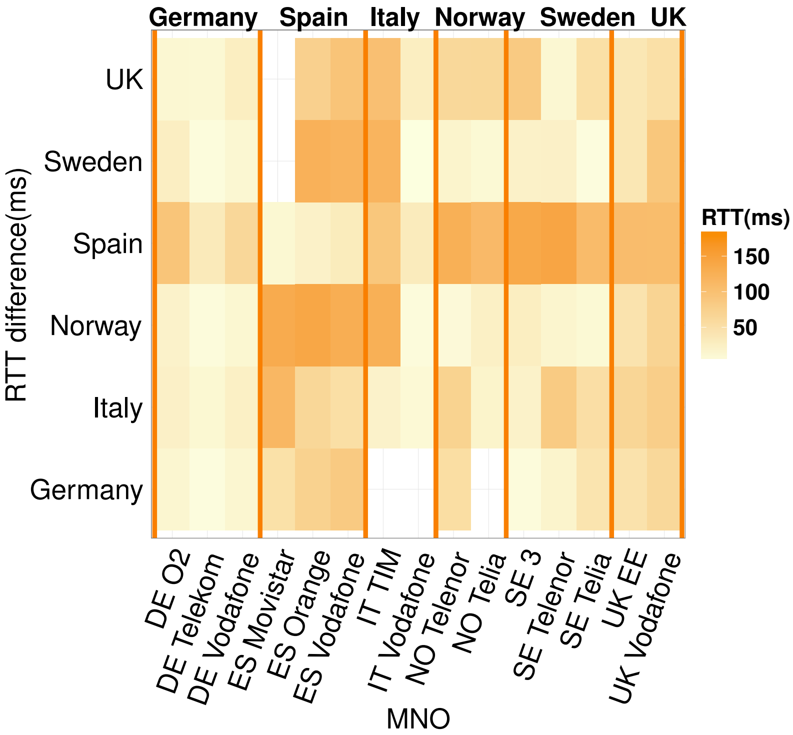


Figure 10.7: RTT difference per operator.

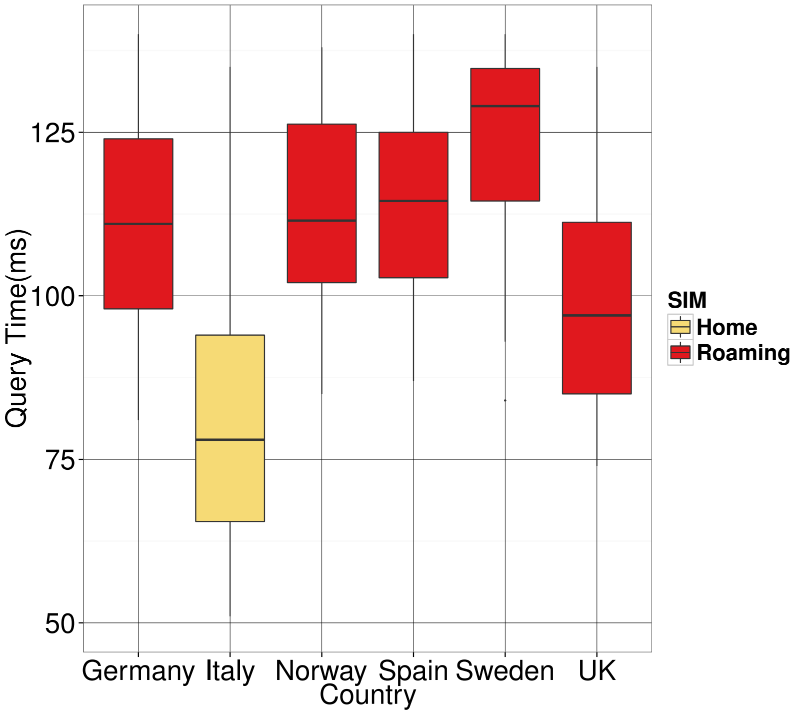


Figure 10.8: DNS Query time to all FQDNs for TIM IT.

between each SIM and the target web servers. The median value of the handshake time from the visited SIMs towards all the targets we measure is 170ms, while the median value for the roaming SIMs is 230ms. This leads to a delay penalty of approximately 60ms. As in the cases before, some MNOs are affected more by this roaming effect than others.

10.4 VoIP

This Section investigates potential traffic differentiation policies (such as blocking or throttling) that may hamper VoIP communications for a roaming user in comparison to a home user. We focus on three popular VoIP applications: FaceTime [13], Facebook Messenger, and Whatsapp. [14]

10.4.1 Experiment Design

We begin with checking whether the MNO allows successful audio/video VoIP calls by carrying out some manual experiments with smart phone running the original applications and making calls among roaming SIMs. This lets us verify possible filtering or blocking in place at the time of the test. If successful, we then check for traffic differentiation mechanisms that could affect the call quality.

The experiment makes three audio and video calls using each application running on a regular mobile phone connected using an instrumented WLAN access point (AP) in our lab. Packet traces were recorded using `tcpdump` resulting in 18 traces, each with duration between [60,80] s. We verify the call-setup phase, which used a complex mix of TCP, STUN [15], and custom protocols to setup the end-to-end communication. From each trace, we then extract the actual audio/video streams. In the next step, we create a Docker container with pre-loaded traces, which we replay using `tcpreplay` to properly edit them so that packets are directed toward dedicated receivers hosted in our premises.

All applications run SRTP [16] on top of UDP, enabling easy adjustment of the packet timing and updating of the source and destination IP addresses. The dedicated server in each country acts as a UDP receiver with a custom signalling TCP connection to log the status of the node (visited network, node identifier, metadata, experiment type, etc.) and the experiment associated with the receiver. In each test, the mobile node sequentially replays the pre-recorded traces with two receivers: a call to a destination in the home country, and a call to a destination in the visited country. For each call, we record `pcap` traces on both the client and server sides. We post-process these

[13] iOS 11: iOS Security Guide. https://www.apple.com/business/docs/iOS_Security_Guide.pdf. [Online; accessed 06-March-2018]

[14] WhatsApp Encryption Overview. https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf. [Online; accessed 06-March-2018]

[15] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing. Session Traversal Utilities for NAT (STUN). RFC 5389, Oct. 2008b. URL https://www.rfc-editor.org/rfc/rfc5389.txt

[16] M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman. The Secure Real-time Transport Protocol (SRTP). RFC 3711 (Proposed Standard), Mar. 2004. ISSN 2070-1721. URL https://www.rfc-editor.org/rfc/rfc3711.txt

traces to check for traffic differentiation.

10.4.2 VoIP Results

We first verify that calls can be freely made from the tested network. We found that all operators allow users (even when roaming) to freely make Voice over IP (VoIP) calls using popular applications on their smart-phone. This confirms that no filtering was in place at the time of testing.

Next, we consider eventual traffic differentiation. We analyze well-known QoS metrics for real-time VoIP applications: packet loss, instantaneous bit rate, and Inter-Packet Gap (IPG), the time difference between two consecutive packets, to detect traffic differentiation. The results show that the packet loss ratio is less than 1% in all experiments. We conclude that no operators introduce artificial packet loss during our tests. MNOs could do so if desired to reduce the quality of calls for these applications and enforce traffic policing.

For each trace, we compare the bit rate we observe at the sender side and at the receiver side. Figure 10.9 presents this as an ECDF for the three applications. Solid/dashed lines indicate the sender/receiver side when calling a receiver in the home or visited country (operator O2 DE). The applications use different audio/video codec combinations with different bit rate requirements. We observe no differences when using the tested network and the home network.

Figure 10.9: Bitrate for operator O2 DE.

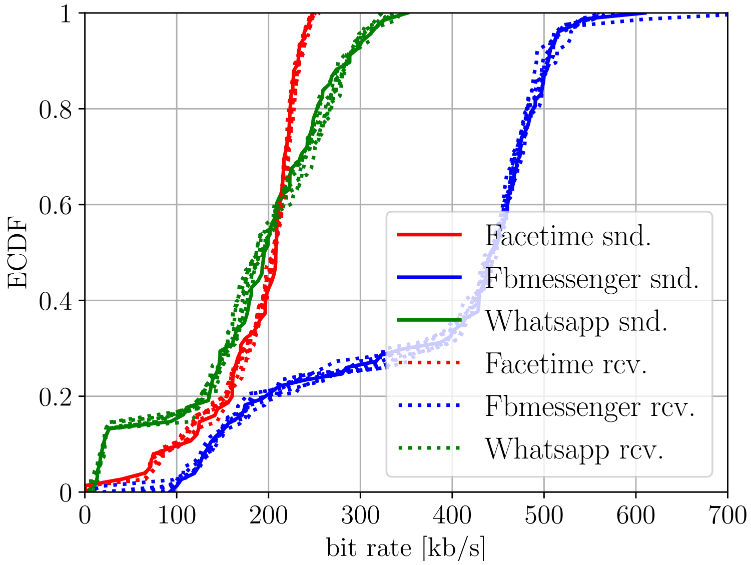


Figure 10.10 shows a mobile user of the O2 DE operator making a Facetime call to our server in Germany while roaming. The periodic 60 ms long IPG is typical of low rate audio codecs the modern VoIP applications use. We observe some differences when comparing measurements at the sender (solid line) and the receiver (dashed lines – one for each visited country). Some gaps are compressed (a smaller IPG), while others become expanded (a larger IPG). We observe this in all experiments with all operators when the sender is in its home country. We ascribe this to the modulation of the IPG by 3G/4G access mechanisms. Given the IPG is bounded to less than 80 ms, we conclude that this would not hamper voice quality, and expect these variations to be absorbed by the receiver playout buffer. [17]

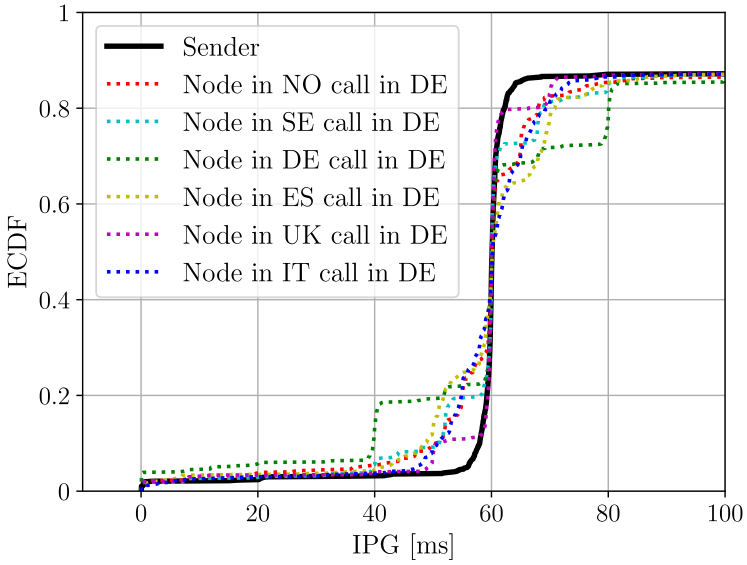


Figure 10.10: IPG for operator O2 DE.

All experiments give very similar results, pointing to no evidence of traffic manipulation. We summarize these findings using the well-known KS Test [18] and P-Value [19] to determine whether the ECDFs differ between the sender (our reference) and the receiver. If statistically similar, the KS would be close to 0 while P-Value would be close to 1. If significantly different, the KS would be greater than 0, and the P-Value close to 0. Figure 10.11 shows the scatter plot of the (KS, P-Value) points. Our results confirm that the receiver throughput is statistically identical to the sender throughput in all experiments. The IPG statistics are affected by the 3G/4G access mechanisms that alter the distribution (albeit not impairing the VoIP quality).

[17] R. Birke, M. Mellia, M. Petracca, and D. Rossi. Experiences of voip traffic monitoring in a commercial ISP. *International Journal of Network Management*, 20(5):339–359, 2010

[18] Kolmogorov–Smirnov Test. The Concise Encyclopedia of Statistics, pages 283–287, 2008. ISBN 978-0-387-32833-1

[19] R. L. Wasserstein and N. A. Lazar. The ASA's Statement on p-Values: Context, Process, and Purpose. *The American Statistician*, 70(2):129–133, 2016. DOI: 10.1080/00031305.2016.1154108

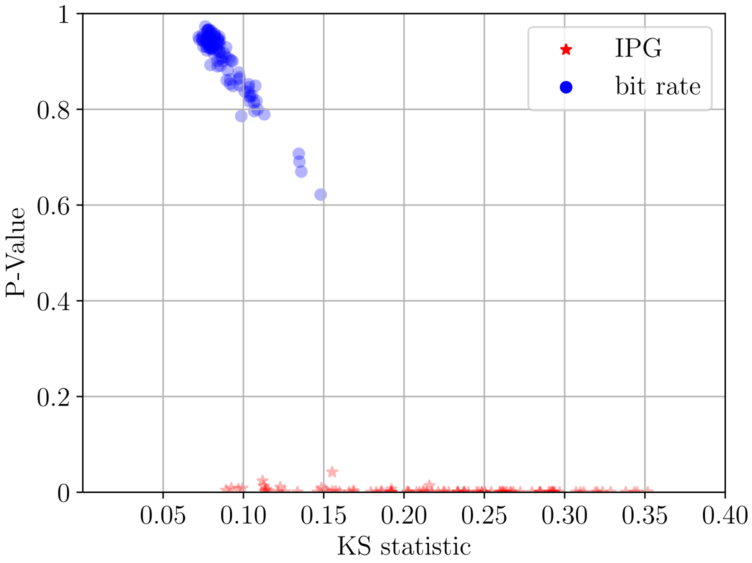


Figure 10.11: KS and P-Value test.

Finally, while QoS in terms of IPG and throughput are good, the total end-to-end delay could be significantly higher when roaming because of the HR solution. The one-way-delay could thus grow excessively large when two roaming SIMs call each other, making the interactive voice conversation difficult. The same effect was faced in GSM networks, and fixed by anti-tromboning [20] solution (e.g., allowing local breakout for voice traffic).

Takeaway We do not observe any traffic differentiation on any of the 16 MNOs we measure. However, the additional delay of HR could impair real-time applications. This is an old issue (typically referred to as tromboning) which has been solved in GSM networks, but persists for 3G/4G VoIP applications.

10.5 Content Discrimination

In this Section, we evaluate the availability of content when roaming, in particular whether MNOs filter website content and apply geographical restrictions. There are many reasons operators could have content filtering, which include complying with government guidelines or following court orders, e.g, to restrict access to file-sharing websites in the UK [21], or the use of ’opt-out’ parental filters. We refer to any differences in the availability of websites and their content due to network interference as "content discrimination". When this dif-

[20] C. Aoun. Identifying intra-realm calls and Avoiding media tromboning. Internet-Draft draft-aoun-midcom-intrarealmcalls-00, Internet Engineering Task Force, Feb. 2002. URL `https://datatracker.ietf.org/doc/html/draft-aoun-midcom-intrarealmcalls-00`. Work in Progress

[21] E. Rosati. 2015: the year of blocking injunctions? *Journal of Intellectual Property Law & Practice*, 10(3):147, 2015

ference is attributed to geographical location rather than the studied network, it is known as "content geo-restriction".

10.5.1 Experiment Design

The OONI[22] provides software tests for detecting censorship, surveillance and traffic manipulation in the Internet, using open source software. We ran two measurement campaigns using OONI's tool `ooniprobe` to detect network interference in home and roaming scenarios, geared at both content discrimination and content geo-restriction.

The `ooniprobe` web connectivity test [23] performs the following steps over both the network of interest (tested network, using both home and roaming SIMs) and the Tor network [24]: resolver identification, DNS lookup, TCP connect and HTTP GET requests. First, ooniprobe performs DNS queries to disclose the IP endpoint of the DNS resolver in the tested network, and records the response, alongside a control response returned by Google's DNS resolver. Then, a TCP connection on port 80 (or port 443 for URLs that support TLS) is attempted using the list of IP endpoints identified in the DNS responses. HTTP GET requests are sent towards a list of URLs over both the tested network and over the Tor network and the responses are recorded. Differences in the results for the two tests are indicative of network interference. The results are made available to the public via the OONI API[25]. Our first set of tests considers 50 randomly selected websites from `ooniprobe`'s default global censorship list Citizenlab [2018] contributed by users, seeking to identify content discrimination for roaming users. For the second set of measurements, we provide a list of 15 websites known to be available locally in the tested countries, but geo-restricted abroad. Beside network interference profile matching for the home and visited results, we additionally searched HTTP responses for known geo-restriction indicators and warnings. The high latency of the Tor network and data quota limitations limited us to test only a small number of websites.

10.5.2 Content Discrimination Results

We create a network interference profile for each measurement, containing the names of blocked websites and the HTTP body responses. For each visited website, we compare the responses observed using the plain connectivity against the one obtained using the TOR network. If those are different, this is an indication of manipulation. We then consider the home country interference profile as the baseline profile against which we compare the roaming profiles for that MNO to check if additional filtering is in place when roaming. If the HTTP

[22] https://ooni.torproject.org/

[23] Ooni - ooni: Open observatory of network interference. URL `https://ooni.torproject.org/`

[24] K. Loesing, S. J. Murdoch, and R. Dingledine. A case study on measuring statistical data in the Tor anonymity network. In *Proceedings of the Workshop on Ethics in Computer Security Research (WECSR 2010)*, LNCS. Springer, January 2010

[25] https://api.ooni.io/

	UK	NO	SE	DE	IT	ES
Vodafone UK	100%	98%	99%	98%	99%	86%
Telenor NO	99%	100%	98%	98%	98%	97%
Telia SE	99%	97%	100%	99%	99%	99%
Vodafone DE	98%	97%	99%	100%	98%	97%
TIM IT	98%	98%	98%	97%	100%	97%
Orange ES	99%	98%	98%	N/A	97%	100%

Table 10.3: Interference profile match Home vs. Roaming. High percentage indicate no difference between home and roaming setup.

response was unavailable due to censorship, we record the blocking mechanism as reported by ooniprobe.

We tested a mixture of dynamic and static websites and saw no content discrimination, with a perfect match of 95% and above of the tests between the home and roaming case. That is, if any censorship or blocking is present, it is the same in home and roaming case (this is consistent with HR again). The detailed results (shown in Table 10.3) are consistent with HR for all MNOs. The only exception is the case for Vodafone UK, where 3 websites were blocked (by DNS) in the home country, but were instead available in Spain. We double checked this, and found them to be a measurement artefact where ooniprobe incorrectly reported the blocking method, but we could not reproduce it later. We note N/A in the table where the results could not be collected at the time of measurement.

The findings are similar for the geo-restricted content tests. Here we check if a geo-restricted content that is accessible from home could be accessed also when roaming. To detect this, we search the body of the websites for known content indicating geo-restriction. We found results consistent with HR: content available in a user's home country remains available when roaming. Table 10.4 presents the profile match between the roaming profile and that of the visited network, for geo-restricted content. We did not observe substantial differences between the content policy of the two scenarios. We did also further (manual) investigation to verify video streaming application behaviour. We observed 10 cases where the website was accessible with no limitations, but then the actual video streaming was blocked with an alert to the user on the restriction.

It is worth noting that different countries may have different interference profiles. This can be due to, e.g., different court rules or legal

	UK	NO	SE	DE	IT	ES
EE UK	100%	86%	89%	80%	87%	88%
Telenor NO	93	100%	94%	90%	88%	90 %
Telia SE	N/A	88%	100%	92%	90%	89%
O2 DE	87%	89%	82%	100%	N/A	84%
TIM IT	N/A	85%	83%	85%	100%	87%
Orange ES	N/A	85%	83%	N/A	89%	100%

Table 10.4: Interference profile match for geo-restricted websites, Roaming vs. Visited. Differences are higher when comparing accessible content as home user or as roaming user.

guidelines. Given HR, a roaming user is always subject to his home country rules, even when visiting a foreign country where different laws are in place. This justifies the slightly lower figures in Table 10.4.

Takeaway We found no evidence of content discrimination and geo-restriction for users in roaming scenarios, and the experience of browsing websites was the same in roaming and at home. However, there are clear differences between experience of a user of a network and a user visiting the same network, including the inability to retrieve geo-restricted content and the availability of different content.

10.6 Discussion

10.6.1 On Measurements Limitations

The MONROE-Roaming platform integrates measurement nodes located in six different EU countries and a measurement responder per country. Although this allows us to capture at an unprecedented scale the performance of international roaming in Europe, it is still a limited view in terms of spatial sample distribution within each country (we only use two hardware devices per country, in the same location). Similarly, the findings in this paper refer to just one snapshot – and the community should repeat these experiments over time to identify and investigate changes. The high cost of mounting such an experimental study is a major restricting factor for the density of sampling geo-locations. We instead focus on characterizing multiple MNOs by taking advantage of the SIM farm we built using MONROE-Roaming. For each MNO, we purchased a similar data plan (10GB/month) enabling us to capture similar number of samples per MNO and country. Furthermore, using the same equipment type throughout the measurement platform and in all locations eliminates potential device bias we might observe in the measurement samples.

Our measurement study focuses mainly on network performance and content implications of the roaming solutions in Europe. We leave for future work the exploration of potential performance penalties (see Section 10.3) on actual end-user Quality of Experience (QoE).

10.6.2 On Roaming Configurations

LBO appears a natural choice for an IP-based service and could offer lower operational cost as well as cheaper data tariffs. At the same time, we have shown that this can eliminate delay and potentially increase capacity for some traffic (depending on the destination). Although, LBO relies on access to local infrastructure, offering this could act as a product differentiator for the MNOs that provide this service first. In contrast, HR provides the home MNOs with all the accounting and billing information. This has been verified to be the major problem with MNOs that need to have near real-time view of the customer traffic for accounting reasons.

Whereas Session Initiation Protocol (SIP) signalling could be used to derive billing information for voice (and VoLTE) calls, an MNO typically uses records to issue bills. Breakout at different points complicates this accounting, with possible abuse from customers (e.g, the delay in billing might allow excessive amount of data traffic when roaming). Within the cellular network, classes of traffic can be differentiated using the Access Point Name (APN) and QoS Class Identifier (QCI). This could be used, for instance, by MNOs to implement HR for data, but LBO for VoLTE. [26] This raises the question of whether the roaming agreement could be updated using the same principle to break out some/all data traffic.

Any additional complexity from LBO can add to the operational cost of supporting users of the network (e.g., debugging issues, tracking faults, and predicting traffic). And if a service fails, it is not obvious who is responsible for finding the fault and fixing this. An IP Packet Exchange (IPX) can help mitigate these impacts. Some solutions introduce additional proxy elects [27], responsible for routing traffic towards the correct network, and the associated control functions to coordinate.

Additionally, there are filtering rules, Digital Rights Management (DRM), language preference and personal content that depend upon the location (country) in which the content is viewed. Lawful intercept further complicates the picture. Here, the home network has

[26] I. Kaltsas. Make Yourself at Home : A Comparative Study of VoLTE Roaming Architectures. Master's thesis, KTH, School of Information and Communication Technology (ICT), 2017

[27] GSM Association: Guidelines for IPX Provider Networks. https://www.gsma.com/newsroom/wp-content/uploads//IR.34-v13.0-1.pdf, a. [Online; accessed 06-March-2018]

full visibility of the necessary data, but the visited network may not. Lawful intercept may be further complicated because of variations in regulatory requirements depending on the geographic location of equipment. In a nutshell, enforcing and accounting for multiple policies for different content in different locations can become complex. Home routing simplifies this by letting the original operator see and manage all the traffic.

Lastly, access to content served by Content Delivery Networks (CDN) needs to be carefully optimized to avoid cases where a roaming user is redirected to local replica that is spatially close, but whose network path is unnecessarily long (due to breakout constraints).

The choice of which form of roaming is used therefore is a function of the roaming agreement and capabilities of the visited network. These are constrained by many technical and legal requirements. Therefore, different breakout options can affect performance of application in different ways.

11 Conclusions and Future Work

In this work we describe an experimental model for using crowdsourcing platforms to perform large-scale Internet measurements. Our research efforts expand the traditional crowdsourcing focus from the human element to use a diverse and numerous group of end-user devices as measurement vantage points and use crowdsourcing large scale measurements platforms. We demonstrate the described approach while assessing the feasibility of deploying encryption by default in the Internet.

We focus our crowdsourcing campaigns on building a representative dataset to show the potential success of widespread adoption of TLS encryption for existing protocols and HTTP2 in their native ports. We argue that the proposed experimental setup gives us a realistic idea on the behavior of the Internet ecosystem towards the deployment of the secure versions of protocols using different ports. In this context, we exemplify how to overcome several of the limitations of the crowdsourcing platforms, including the collection of specific user data without having direct control over the measurement agents.

We find that in average the failure rate of TLS over different ports is near the 6%. We also find that in the case of mobile networks where proxies are used, the failure rate can be as high as 70%. We conclude that it is probably feasible to roll out TLS protection for most ports except for port 80, assuming a low failure rate (6%). We argue that probably port 80 is the one port where using TLS is less needed, as there is already in place a well known mechanism to secure web communication through port 443.

For HTTP2, we tested 4 HTTP/2 variations accessing 67 different ports from 38 countries world-wide. Our preliminary results suggest that the presence of middleboxes in the Internet can detract from H2C deployment, especially if the server is listening on port 80 and the client resides on a mobile network. In fact, the Upgrade mechanism failed up to a 7% of the times.

We also believe that our results can serve as a lower bound for the failure rate for using protocols other than expected in different ports.

We then worked on extending our tool to also test new protocols at transport layer, like TCP Fast Open and ECN++.

We evaluate the interaction of TFO with the elements of the path. Lessons learned from this study can help in designing robust TCP protocol extensions in the presence of middleboxes. Only the 41,3% of the paths we test allowed for a successful TFO communication. Once the client is able to receive the cookie, the 32,14% of the packets are received with no data in the SYN. In the case in which we force data in the SYN without the TFO option, the percentage of success is 50%.

Testing ECN++, we found good news for the deployment of ECN++, which was the original subject of the study. And we can confirm that ECN adoption is proceeding well over fixed networks. However, we found bad news over mobile. More than half of the 18 mobile carriers tested routinely wipe the ECN field of all incoming packets. Fortunately, wiping the ECN field at the first hop only denies the benefits of ECN to the connection; the session otherwise proceeds as normal.

ECN problems in mobile have not surfaced before because this is the first ECN study to have extended to a broad enough set of mobile vantage points. This is due to the considerable work needed to build infrastructure like MONROE, which makes it feasible to measure the effect of kernel level changes over a wide range of mobile networks.

We conjecture that wiping ECN could be due to a bug, where wiping the Diffserv field accidentally includes ECN. Similar bugs have been fixed quickly in the past. We could not prove any correlation between ECN and Diffserv wiping, given we found Diffserv is always wiped. We plan further work on this, in cooperation with the affected carriers.

We then argue that evaluating how the middleboxes behave and particurlarly how NATs behave can be useful for future protocols design. Part of our work is devoted to evaluate the behavior of NATs in fixed and mobile lines. We presented NATwatcher, a tool that allows to shed some light on the NAT behavior in the wild. Using a crowdsourcing approach we provide insight into how vendors, ISPs or users configure and use NATs with respect to TCP, UDP and ICMP packets, providing useful data for designing future applications. We presented the first large-scale measurement campaign of home NATs, based on data from 781 homes, from 65 countries and from 280 ISPs. The data we collected are good enough for understanding NATs deployed behaviors, but not necessarily statistically representative of the Internet that is composed by billions of devices.

Our study demonstrated that about 80% of the tested NATs follow the IETF sanctioned behavior with respect to 11 tests over 17

(64% of tests) For the remaining 6 tests our findings show that only 13% of the NAT boxes follow the IETF requirements of filtering, hairpinning and mapping lifetime over 2 minutes. Moreover, we listed the 11 most common configurations, finding that the 52.5% of the tested NAT boxes use Endpoint-Independent mapping for UDP and TCP and Address and port dependent filtering, they do not support hairpinning or UDP mapping over 2 minutes. To the best of our knowledge, this is the largest dataset available describing the behavior of the deployed NAT base and we believe it would provide useful input for application and protocol designers aiming to make their applications and protocols to work across NATs.

On the other hand, despite concerns about its performance impact, CGN solutions are part of the technology landscape during this ongoing phase of transition from IPv4 to IPv6. Though passive data analysis allowed the quantification of CGN penetration in the current Internet ecosystem, we have no knowledge of a client-size only tool that can empower the end-user to determine the upstream NAT configuration. Revelio aims to fill this gap. Using NAT Revelio, we conducted a large-scale active measurement study for CGN detection targeting Europe and the U.S. More specifically, we deployed Revelio on two hardware-based crowdsourcing measurement platforms: RIPE Atlas (with significant presence in Europe) and FCC's Measuring Broadband America (with significant presence in the United States). In total, we instrumented 5,121 measurement vantage points in over 60 different ISPs, capturing myriad combinations of home network topologies and Internet access technologies. We found that 10% (6 out of 64) of the ISPs we tested have some form of CGN deployment. We believe the combination of the FCC and RIPE Atlas study represents the largest active measurement study of CGN deployment in broadband networks to date.

We also validated our results at the probe level with representatives of 4 of the ISPs we tested (approx. 10%). Considering the ground truth we collected, we calculate the accuracy of our method at 100%. However, due to the limitation of the methodology, there are cases when some of the Revelio tests cannot run, thus hindering the test-suite efficiency. Overall, in 24% of the lines we tested, Revelio gave inconclusive results. In the Atlas platform, where the *probes* do not support UPnP, the rate of inconclusive results is 36%. This decreases significantly to only 12% of the *SK probes* in the FCC-MBA testbed, where Revelio was able to invoke UPnP actions.

In order to better understand CGN in mobile network, we then created a measurement test suite for the analysis of CGN in cellular network. While the experiments were designed to run in the MONROE nodes, the test suite can also be used on other measurement platforms. Moreover, our container is easily extendable. During the development of the test suite we gained experience with the MONROE testbed. The results show different configuration between MNOs. In general, the mapping timeouts are lower than recommended by the RFCs. Our results showed similar behaviour for TCP and UDP.

We finally studied CGNs in roaming scenario, discovering interesting insides on roaming configuration. Different roaming network configuration options can indeed affect performance of various applications for the end user. In practice, although there are three possible solutions (i.e., HR, LBO or IHBO), we find that HR is the norm for the MNOs we measured. This comes with performance penalties on the roaming user, who experiences increased delay and appears to the public internet as being connected in the home country. This has further implications in the selection of CDN server replica when roaming abroad, because the mobile user will access a server in the home network rather than one close to their location. However, at the same time, the roaming user is still able to access (in majority of cases) the geo-restricted services from the home country in her native language.

We put these results in perspective while trying to also speculate on the commercial implications of the "Roam like Home" initiative. As regulation reduces the ability of MNOs to compete on price, the subscribers' quality of experience will potentially become a key factor in choosing a provider. The subscribers will increasingly start to compare the roaming experience to the home experience. Thus, an expectation of high quality, always-on services in a visited network follows and if a home network fails to deliver in the visited network, the risk of churn increases. To this end, LBO is a natural step for an IP-based service, and could offer lower operational cost, and cheaper tariffs for data, while at the same time we have shown this can eliminate delay and potentially increase capacity for some traffic (dependent on the destination). Although LBO relies on access to the infrastructure of the visited network which can have implications on service control and charging, offering this could act in the advantage of the first operators to provide the service. Furthermore, in some cases, under the "Roam like Home" paradigm, some users may purchase SIMs from abroad to use in their country under permanent roaming conditions.

References

Measuring Broadband America. https://www.fcc.gov/general/measuring-broadband-america. p. 90

Method and System For Hub Breakout Roaming. https://patents.google.com/patent/US20140169286/en. [Online; accessed 06-March-2018]. p. 114

European Commission: New Rules on Roaming Charges and Open Internet. https://ec.europa.eu/digital-single-market/en/news/new-rules-roaming-charges-and-open-internet. [Online; accessed 06-March-2018]. pp. 8 and 113

Potaroo Network Tool. http://www.potaroo.net/tools/ipv4/. Accessed on 2019-02-02. p. 5

iOS 11: iOS Security Guide. https://www.apple.com/business/docs/iOS_Security_Guide.pdf. [Online; accessed 06-March-2018]. p. 126

GSM Association: Guidelines for IPX Provider Networks. https://www.gsma.com/newsroom/wp-content/uploads//IR.34-v13.0-1.pdf, a. [Online; accessed 06-March-2018]. pp. 113 and 133

GSM Association: IPX White Paper. https://www.gsma.com/iot/wp-content/uploads/2012/03/ipxwp12.pdf, b. [Online; accessed 06-March-2018]. p. 113

GSM Association: LTE and EPC Roaming Guidelines. https://www.gsma.com/newsroom/wp-content/uploads/IR.88-v15.0.pdf, a. [Online; accessed 06-March-2018]. p. 114

Huawei: LTE International Roaming Whitepaper. http://carrier.huawei.com/en/technical-topics/core-network/LTE-roaming-whitepaper, b. [Online; accessed 06-March-2018]. p. 114

Ooni - ooni: Open observatory of network interference. URL https://ooni.torproject.org/. p. 130

Samknows platform. https://www.samknows.com/. Accessed on 2019-02-03. p. 2

UPnP Forum. UPnP Specifications. http://upnp.org/sdcps-and-certification/standards/. Accessed: 2016-06-17. p. 85

WhatsApp Encryption Overview. https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf. [Online; accessed 06-March-2018]. p. 126

Kolmogorov–Smirnov Test. The Concise Encyclopedia of Statistics, pages 283–287, 2008. ISBN 978-0-387-32833-1. p. 128

https://www.ietf.org/mailman/listinfo/hops. *IRTF*, 2015. pp. 1 and 20

https://github.com/square/okhttp. *OkHttp - An HTTP+SPDY client for Android and Java application*, 2015a. p. 30

Is the web HTTP/2 yet? *http://isthewebhttp2yet.com/*, 2015b. p. 24

https://github.com/square/okhttp/issues/1305. *Don't use broken ALPN on Android 4.4 | GitHub*, 2015. p. 28

http://www.speedguide.net/port.php?port=593. *Port 593 filter*, 2015. p. 39

MONROE: Git repository, 2017. https://github.com/monroe-project/. p. 58

Ö. Alay, A. Lutu, R. García, M. Peón-Quirós, V. Mancuso, T. Hirsch, T. Dely, J. Werme, K. Evensen, A. Hansen, et al. Measuring and Assessing Mobile Broadband Networks with MONROE. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2016 IEEE 17th International Symposium on A*, pages 1–3. IEEE, 2016. p. 58

O. Alay, A. Lutu, M. Peón-Quirós, V. Mancuso, T. Hirsch, K. Evensen, A. Hansen, S. Alfredsson, J. Karlsson, A. Brunstrom, A. Safari Khatouni, M. Mellia, and M. A. Marsan. Experience: An open platform for experimentation with commercial mobile broadband networks. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, MobiCom '17, pages 70–78, New York, NY, USA, 2017a. ACM. ISBN 978-1-4503-4916-1. pp. 2 and 15

Ö. Alay, A. Lutu, M. P. Quirós, V. Mancuso, T. Hirsch, K. Evensen, A. F. Hansen, S. Alfredsson, J. Karlsson, A. Brunstrom, A. S. Khatouni, M. Mellia, and M. A. Marsan. Experience: An Open Platform for Experimentation with Commercial Mobile Broadband Networks. MobiCom 2017, pages 70–78, 2017b. DOI: 10.1145/3117811.3117812. pp. 14 and 115

M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data Center TCP (DCTCP). *Proc. ACM SIGCOMM'10, Computer Communication Review*, 40(4): 63–74, Oct. 2010. pp. 51 and 54

C. Aoun. Identifying intra-realm calls and Avoiding media tromboning. Internet-Draft draft-aoun-midcom-intrarealmcalls-00, Internet Engineering Task Force, Feb. 2002. URL https://datatracker.ietf.org/doc/html/draft-aoun-midcom-intrarealmcalls-00. Work in Progress. p. 129

F. Audet and C. Jennings. Network Address Translation (NAT) Behavioral Requirements for Unicast UDP. IETF, RFC 4787, January 2007. pp. 7, 66, and 76

M. Bagnulo and B. Briscoe. ECN++: Adding Explicit Congestion Notification (ECN) to TCP Control Packets. Internet Draft draft-bagnulo-tcpm-generalized-ecn-04, Internet Engineering Task Force, May 2017. URL https://tools.ietf.org/html/draft-bagnulo-tcpm-generalized-ecn. (Work in Progress). pp. 4 and 52

V. Bajpai, S. J. Eravuchira, and J. Schönwälder. Lessons learned from using the ripe atlas platform for measurement research. *SIGCOMM Comput. Commun. Rev.*, 45(3):35–42, July 2015. ISSN 0146-4833. p. 2

F. Baker. Requirements for IP Version 4 Routers. RFC 1812 (Proposed Standard), June 1995. URL http://www.ietf.org/rfc/rfc1812.txt. Updated by RFCs 2644, 6633, accessed on 2017-11-27. p. 57

S. Bauer, R. Beverly, and A. Berger. Measuring the State of ECN Readiness in Servers, Clients, and Routers. In *Proc ACM SIGCOMM Internet Measurement Conference (IMC'11))* ,, pages 171–180, 2011. pp. 11 and 51

M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman. The Secure Real-time Transport Protocol (SRTP). RFC 3711 (Proposed Standard), Mar. 2004. ISSN 2070-1721. URL https://www.rfc-editor.org/rfc/rfc3711.txt. p. 126

M. Belshe, R. Peon, and M. Thomson. Hypertext Transfer Protocol Version 2 (HTTP/2). Technical report, RFC 7540, May, 2015. p. 3

P. Bhooma. TCP ECN: Experience with enabling ECN on the Internet. 98th IETF MAPRG Presentation, 2017. URL https://www.ietf.org/proceedings/98/slides/slides-98-maprg-tcp-ecn-experience-with-enabling-ecn-on-the-internet-padma-bhooma-00.pdf. pp. 11, 12, and 51

R. Birke, M. Mellia, M. Petracca, and D. Rossi. Experiences of voip traffic monitoring in a commercial ISP. *International Journal of Network Management*, 20(5):339–359, 2010. p. 128

E. Bocchi, A. S. Khatouni, S. Traverso, A. Finamore, M. Munafò, M. Mellia, and D. Rossi. Statistical network monitoring: Methodology and application to carrier-grade nat. *Computer Networks*, 107:20 – 35, 2016. ISSN 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2016.06.018. URL http://www.sciencedirect.com/science/article/pii/S1389128616301980. Machine learning, data mining and Big Data frameworks for network monitoring and troubleshooting. p. 13

B. Briscoe, M. Kühlewind, and R. Scheffenegger. More Accurate ECN Feedback in TCP. Internet Draft draft-ietf-tcpm-accurate-ecn-02, Internet Engineering Task Force, Oct. 2016. URL http://tools.ietf.org/html/draft-ietf-tcpm-accurate-ecn. (Work in Progress). p. 54

B. Briscoe (Ed.), K. De Schepper, and M. Bagnulo. Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture. Internet Draft draft-ietf-tsvwg-l4s-arch-00, Internet Engineering Task Force, Apr. 2017. URL https://tools.ietf.org/html/draft-briscoe-tsvwg-l4s-arch. (Work in Progress). pp. 51 and 54

M. Buhrmester, T. Kwang, and S. D. Gosling. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 6(1):3–5, 2011. p. 2

T. Burbridge. Personal Communication, 2016. p. 89

S. Cheshire. Networking for the Modern Internet. URL: https://developer.apple.com/videos/play/wwdc2016/714/, June 2016. (Presentation video: requires Safari browser). pp. 5 and 12

B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman. PlanetLab: An Overlay Testbed for Broad-coverage Services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12, July 2003a. ISSN 0146-4833. DOI: 10.1145/956993.956995. p. 59

B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman. Planetlab: An overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12, July 2003b. ISSN 0146-4833. p. 2

Citizenlab. citizenlab/test-lists, Mar 2018. URL https://github.com/citizenlab/test-lists. p. 130

G. Detal, B. Hesmans, O. Bonaventure, Y. Vanaubel, and B. Donnet. Revealing Middlebox Interference with Tracebox. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 1–8, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1953-9. DOI: 10.1145/2504730.2504757. p. 56

A. Doan et al. Crowdsourcing systems on the world-wide web. *ACM*, 2011. p. 2

C. Donley, L. Howard, V. Kuarsingh, J. Berg, and J. Doshi. Assessing the Impact of Carrier-Grade NAT on Network Applications. RFC 7021, September 2013. pp. 6, 12, and 81

A. B. Downey. Using Pathchar to Estimate Internet Link Characteristics. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '99, 1999. p. 85

S. Farrell et al. Pervasive monitoring is an attack. Technical report, RFC 7258, May, 2014. p. 3

S. Floyd. TCP and Explicit Congestion Notification. *ACM SIGCOMM Computer Communication Review*, 24(5):10–23, Oct. 1994. (This issue of CCR incorrectly has '1995' on the cover). p. 52

S. Floyd. Inappropriate TCP Resets Considered Harmful. RFC 3360 (Best Current Practice), Aug. 2002. URL `http://www.ietf.org/rfc/rfc3360.txt`. Accessed on 2017-11-27. p. 51

M. Ford, M. Boucadair, A. Durand, P. Levis, and P. Roberts. Issues with IP Address Sharing. RFC 6269, June 2011. pp. 6, 12, and 81

S. M. Garland and D. B. Smith. Communications Between Service Providers and Customer Premises Equipment, Dec. 26 2000. US Patent 6,167,042. p. 84

U. Goel, M. Steiner, M. P. Wittie, M. Flack, and S. Ludin. Measuring What is Not Ours: A Tale of 3^{rd} Party Performance. In M. A. Kaafar, S. Uhlig, and J. Amann, editors, *Passive and Active Measurement*, pages 142–155. Springer International Publishing, 2017. p. 119

F. Gont et al. Transmission and processing of IPv6 options. Technical report, IETF draft-gont-6man-ipv6-opt-transmit-01, 2015. p. 1

S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. IETF, RFC 5382, October 2008. pp. 7, 66, 76, and 77

M. Handley. Why the internet only just works. *BT Technology Journal*, 2006. pp. 1 and 20

S. Hätönen, A. Nyrhinen, L. Eggert, S. Strowes, P. Sarolahti, and M. Kojo. An Experimental Study of Home Gateway Characteristics. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 260–266. ACM, 2010. p. 12

B. Hesmans et al. Are TCP extensions middlebox-proof? In *Workshop on Hot topics in middleboxes and network function virtualization*. ACM, 2013. p. 1

M. Hirth, T. Hoßfeld, and P. Tran-Gia. Anatomy of a crowdsourcing platform - using the example of microworkers.com. In *2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 322–329, June 2011. DOI: 10.1109/IMIS.2011.89. p. 2

M. Hirth, T. Hossfeld, M. Mellia, C. Schwartz, and F. Lehrieder. Crowdsourced network measurements. *Computer Networks*, 90 (C):85–98, Oct. 2015. ISSN 1389-1286. pp. 11 and 12

M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda. Is it still possible to extend TCP? In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 181–194. ACM, 2011. pp. 1, 4, 11, 12, and 52

J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck. An in-depth study of LTE: effect of network protocol and application behavior on performance. *ACM SIGCOMM Computer Communication Review*, 43(4):363–374, 2013. p. 14

C. Jennings. Nat Classification Test Results. IETF Internet Draft draft-jennings-behave-test-results-04, July 2007. pp. 12 and 72

D. B. K. Ramakrishnan, S. Floyd. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3360 (Best Current Practice), Sept. 2001. URL `https://tools.ietf.org/html/rfc3168`. Accessed on 2019-02-02. pp. 51, 52, 53, and 57

I. Kaltsas. Make Yourself at Home : A Comparative Study of VoLTE Roaming Architectures. Master's thesis, KTH, School of Information and Communication Technology (ICT), 2017. p. 133

F. Kaup, F. Michelinakis, N. Bui, J. Widmer, K. Wac, and D. Hausheer. Assessing the Implications of Cellular Network Performance on Mobile Content Access. *IEEE Transactions on Network and Service Management*, 13(2):168–180, 2016. p. 14

A. S. Khatouni, M. Mellia, M. A. Marsan, S. Alfredsson, J. Karlsson, A. Brunstrom, O. Alay, A. Lutu, C. Midoglu, and V. Mancuso.

Speedtest-like measurements in 3g/4g networks: The monroe experience. In *Teletraffic Congress (ITC 29), 2017 29th International*, volume 1, pages 169–177. IEEE, 2017. p. 14

C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson. Netalyzr: Illuminating the Edge Network. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, pages 246–259. ACM, 2010. p. 13

F. Krueger and S. R. Andrews. Bismark: a flexible aligner and methylation caller for Bisulfite-Seq applications. *Bioinformatics*, 27(11): 1571–1572, 04 2011. ISSN 1367-4803. p. 2

M. Kühlewind, S. Neuner, and B. Trammell. On the state of ecn and tcp options on the internet. In M. Roughan and R. Chang, editors, *Passive and Active Measurement*, pages 135–144, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. pp. 11 and 12

A. Kuzmanovic, A. Mondal, S. Floyd, and K. Ramakrishnan. Adding Explicit Congestion Notification (ECN) Capability to TCP's SYN/ACK Packets. RFC 5562 (Experimental), June 2009. URL http://www.ietf.org/rfc/rfc5562.txt. Accessed on 2017-11-27. p. 57

A. Langley. Probing the viability of TCP extensions. Google Inc., Technical Report: https://www.imperialviolet.org/binary/ecntest.pdf, 2008. p. 11

I. Livadariu, K. Benson, A. Elmokashfi, A. Dhamdhere, and A. Dainotti. Investigating Carrier Grade NAT Deployment Using Passive Measurements. In *submission to the 2017 Network Traffic Measurement and Analysis Conference*, 2017. p. 13

K. Loesing, S. J. Murdoch, and R. Dingledine. A case study on measuring statistical data in the Tor anonymity network. In *Proceedings of the Workshop on Ethics in Computer Security Research (WECSR 2010)*, LNCS. Springer, January 2010. p. 130

A. Lutu, M. Bagnulo, A. Dhamdhere, and K. Claffy. NAT Revelio: Detecting NAT444 in the ISP. In *International Conference on Passive and Active Network Measurement*, pages 149–161. Springer, 2016. pp. 7, 13, and 82

R. Mahy, P. Matthews, and J. Rosenberg. Traversal Using Relay NAT (TURN). IETF, RFC 5766, April 2010. p. 12

A. M. Mandalari, M. Bagnulo, and A. Lutu. TCP Fast Open: Initial Measurements. *ACM CoNEXT Student Workshop, Dec 2015, Heidelberg, Germany.*, 2015a. DOI: 10.1145/2842665.2843561. p. vii

A. M. Mandalari, M. Bagnulo, and A. Lutu. Informing Protocol Design Through Crowdsourcing: The Case of Pervasive Encryption. *ACM SIGCOMM Computer Communication Review*, 45(4):105–123, 2015b. DOI: 10.1145/2787394.2787397. p. vii

A. M. Mandalari, M. A. D. Bautista, F. Valera, and M. Bagnulo. NATwatcher: Profiling NATs in the Wild. *IEEE Communications Magazine*, 55(3):178–185, March 2017a. ISSN 0163-6804. DOI: 10.1109/MCOM.2017.1600776CM. p. viii

A. M. Mandalari, A. Lutu, A. Dhamdhere, M. Bagnulo, and K. Claffy. Tracking the Big NAT across Europe and the U.S. abs/1207.0016, 2017b. URL https://arxiv.org/abs/1704.01296. p. viii

A. M. Mandalari, A. Lutu, B. Briscoe, M. Bagnulo, and O. Alay. Measuring ECN++: Good News for ++, Bad News for ECN over Mobile. *IEEE Communications Magazine*, 56(3):180–186, March 2018a. ISSN 0163-6804. DOI: 10.1109/MCOM.2018.1700739. p. viii

A. M. Mandalari, A. Lutu, A. Custura, A. Safari Khatouni, O. Alay, M. Bagnulo, V. Bajpai, A. Brunstrom, J. Ott, M. Mellia, and G. Fairhurst. Experience: Implications of Roaming in Europe. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, MobiCom '18, pages 179–189, New York, NY, USA, 2018b. ACM. ISBN 978-1-4503-5903-0. DOI: 10.1145/3241539.3241577. p. ix

C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, C. Ziemlicki, and Z. Smoreda. Not All Apps Are Created Equal: Analysis of Spatiotemporal Heterogeneity in Nationwide Mobile Service Usage. CoNEXT 2017, 2017. p. 122

A. Medina, M. Allman, and S. Floyd. Measuring the Evolution of Transport Protocols in the Internet. *SIGCOMM Comput. Commun. Rev.*, 35(2):37–52, Apr. 2005. ISSN 0146-4833. DOI: 10.1145/1064413.1064418. p. 11

D. Merkel. Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux Journal*, 2014(239), Mar. 2014. ISSN 1075-3583. p. 116

F. Michelinakis, H. Doroud, A. Razaghpanah, A. Lutu, N. Vallina-Rodriguez, P. Gill, and J. Widmer. The Cloud that Runs the Mobile Internet: A Measurement Study of Mobile Cloud Services. In *Proc. IEEE INFOCOM*, Honolulu, HI, USA, April 2018. p. 14

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, IMC '15, pages 239–251, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3848-6. p. 14

A. Müller, F. Wohlfart, and G. Carle. Analysis and Topology-based Traversal of Cascaded Large Scale NATs. In *Proceedings of the 2013 Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, 2013. p. 12

D. Naylor, A. Finamore, I. Leontiadis, Y. Grunenberger, M. Mellia, M. Munafò, K. Papagiannaki, and P. Steenkiste. The Cost of the S in HTTPS. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 133–140. ACM, 2014a. p. 23

D. Naylor et al. The cost of the "S" in HTTPS. In *ACM*, CoNEXT, 2014b. p. 3

Y. Ohara, K. Nishizuka, K. Chinen, K. Akashi, M. Kohrin, E. Muramoto, and S. Miyakawa. On the impact of mobile network delays on connection establishment performance of a carrier grade nat device. In *Proceedings of the AINTEC 2014 on Asian Internet Engineering Conference*, AINTEC '14, pages 1:1–1:8, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3251-4. DOI: 10.1145/2684793.2684794. URL http://doi.acm.org/10.1145/2684793.2684794. p. 13

J. Padhye and S. Floyd. On Inferring TCP Behavior. *Proc. ACM SIGCOMM'01, Computer Communication Review*, 31(4):287–298, Oct. 2001. (Aka. Identifying the TCP Behavior of Web Servers). pp. 52, 58, and 61

V. Paxson. Strategies for sound internet measurement. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 263–271. ACM, 2004. p. 26

S. Perreault, I. Yamagata, S. Miyakawa, A. Nakagawa, and H. Ashida. Common Requirements for Carrier-Grade NATs (CGNs). RFC 6888, April 2013. p. 82

J. Postel. Internet Control Message Protocol. RFC 792 (INTERNET STANDARD), Sept. 1981. URL http://www.ietf.org/rfc/rfc792.txt. Updated by RFCs 950, 4884, 6633, 6918, accessed on 2017-11-27. p. 57

S. Radhakrishnan, Y. Cheng, J. Chu, A. Jain, and B. Raghavan. Tcp fast open. In *Proceedings of the Seventh COnference on Emerging Networking EXperiments and Technologies*, CoNEXT '11, pages 21:1–21:12, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-1041-3. p. 45

P. Richter, F. Wohlfart, N. Vallina-Rodriguez, M. Allman, R. Bush, A. Feldmann, C. Kreibich, N. Weaver, and V. Paxson. A Multi-perspective Analysis of Carrier-Grade NAT Deployment. *arXiv preprint arXiv:1605.05606*, 2016. p. 13

E. Rosati. 2015: the year of blocking injunctions? *Journal of Intellectual Property Law & Practice*, 10(3):147, 2015. p. 129

J. Rosenberg. Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/answer Protocols. IETF, RFC 5245, April 2010. p. 12

J. Rosenberg, R. Mahy, P. Matthews, and D. Wing. Session Traversal Utilities for NAT (STUN). RFC 5389, October 2008a. pp. 12 and 84

J. Rosenberg, R. Mahy, P. Matthews, and D. Wing. Session Traversal Utilities for NAT (STUN). RFC 5389, Oct. 2008b. URL `https://www.rfc-editor.org/rfc/rfc5389.txt`. p. 126

J. H. Salim and U. Ahmed. Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks. RFC 2884 (Informational), July 2000. URL `http://www.ietf.org/rfc/rfc2884.txt`. Accessed on 2017-11-27. p. 51

H. Schulzrinne. Location-to-URL Mapping Architecture and Framework. RFC 5582 (Informational), Sept. 2009. URL `http://www.ietf.org/rfc/rfc5582.txt`. p. 61

R. Sherwood, B. Bhattacharjee, and R. Braud. Misbehaving TCP Receivers Can Cause Internet-Wide Congestion Collapse. Tech report UMD-CS-TR-4737, UMD, 2005. p. 52

N. Skoberne, O. Maennel, I. Phillips, R. Bush, J. Zorz, and M. Ciglaric. IPv4 Address Sharing Mechanism Classification and Tradeoff Analysis. *Networking, IEEE/ACM Transactions on*, 22(2): 391–404, April 2014. p. 12

P. Srisuresh, B. Ford, S. Sivakumar, and S. Guha. NAT Behavioral Requirements for ICMP. IETF, RFC 5508, April 2009. pp. 7, 66, and 76

S. Sundaresan, W. De Donato, N. Feamster, R. Teixeira, S. Crawford, and A. Pescapè. Broadband Internet Performance: a View from the Gateway. In *ACM SIGCOMM Computer Communication Review*, volume 41, pages 134–145. ACM, 2011. p. 85

D. Thaler, D. Bansal, and M. Sridharan. Implementation Report on Experiences with Various TCP RFCs. In *Proc. IETF-68: Transport Area Open Meeting*. Internet Engineering Task Force, Mar. 2007. (Presentation slides). p. 51

B. Trammell, M. K'uhlewind, D. Boppart, I. Learmonth, G. Fairhurst, and R. Scheffenegger. Enabling Internet-Wide Deployment of Explicit Congestion Notification. In *In Proc Passive & Active Measurement (PAM'15) Conference*, 2015. pp. 11, 51, and 61

S. Triukose, S. Ardon, A. Mahanti, and A. Seth. Geolocating IP Addresses in Cellular Data Networks. In *Passive and Active Measurement*, pages 158–167. Springer, 2012. pp. 6, 13, and 81

N. Vallina-Rodriguez, S. Sundaresan, C. Kreibich, and V. Paxson. Header Enrichment or ISP Enrichment?: Emerging Privacy Threats in Mobile Networks. In *Proceedings of the 2015 ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, pages 25–30. ACM, 2015a. p. 20

N. Vallina-Rodriguez, S. Sundaresan, C. Kreibich, N. Weaver, and V. Paxson. Beyond the radio: Illuminating the higher layers of mobile networks. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, pages 375–387. ACM, 2015b. p. 14

V. Ververis, G. Kargiotakis, A. Filastò, B. Fabian, and A. Alexandros. Understanding internet censorship policy: The case of greece. In *5th USENIX Workshop on Free and Open Communications on the Internet (FOCI 15)*, Washington, D.C., 2015. USENIX Association. URL https://www.usenix.org/conference/foci15/workshop-program/presentation/ververis. p. 14

Z. Wang, Z. Qian, Q. Xu, Z. Mao, and M. Zhang. An Untold Story of Middleboxes in Cellular Networks. In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, pages 374–385, 2011. ISBN 978-1-4503-0797-0. pp. 6, 13, and 81

R. L. Wasserstein and N. A. Lazar. The ASA's Statement on p-Values: Context, Process, and Purpose. *The American Statistician*, 70(2): 129–133, 2016. DOI: 10.1080/00031305.2016.1154108. p. 128

J. Weil, V. Kuarsingh, C. Donley, C. Liljenstolpe, and M. Azinger. IANA-Reserved IPv4 Prefix for Shared Address Space. RFC 6598, April 2012. pp. 83, 86, and 95

M. Welzl and G. Fairhurst. The Benefits of using Explicit Congestion Notification (ECN). Internet Draft draft-ietf-aqm-ecn-benefits-08, Internet Engineering Task Force, Nov. 2015. URL https://tools.ietf.org/html/draft-ietf-ecn-benefits. (Work in Progress). pp. 4 and 51