Informing Protocol Design Through Crowdsourcing: the Case of Pervasive Encryption

Anna Maria Mandalari
University Carlos III of Madrid, Spain

Marcelo Bagnulo
University Carlos III of Madrid, Spain

Andra Lutu
Simula Research Laboratory, Norway

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 paper, we show how to make informed protocol design choices by using a crowdsourcing platform. We consider a specific use case, namely the case of pervasive encryption in the modern Internet. Given the latest public disclosures of the NSA global surveillance operations, the issue of privacy in the Internet became of paramount importance. Internet community efforts are thus underway to increase the adoption of encryption. Using a crowdsourcing approach, we perform large-scale TLS measurements to advance our understanding on whether wide adoption of encryption is possible in today's Internet.

CCS Concepts

•**Networks** → **Network measurement;**

Keywords

Crowdsourcing; TLS; Internet measurements; Middleboxes

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

C2B(1)D'15, August 17-21, 2015, London, United Kingdom
© 2015 ACM. ISBN 978-1-4503-3539-3/15/08…$15.00
DOI: http://dx.doi.org/10.1145/2787394.2787397

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 [6]. 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 [5]. Morevoer, 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 [1].

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 [8, 7] on middleboxes behavior attempt to provide such information. However, the existing measurements use only a very small number of vantage points, e.g., in [8] 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.

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.

The emerging sea of crowdsourcing (such as the Amazon Mechanical Turk, Microworkers and others) can provide an accessible alternative to perform large scale Internet 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 [3] we can leverage on crowdsourcing platforms to run Internet wide measurements.

In this paper, 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.

The case of pervasive encryption

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 [4]. We have witnessed recently that many popular applications (e.g., web, Youtube video streaming) have migrated from HTTP to the HTTPS protocol [9]. 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.

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 paper, 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.

2. METHODOLOGY

In this section, we describe the measurements methodology we employ to assess the potential success of deploying secure protocols in the Internet using crowdsourcing. 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.

2.1 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.

In this paper, we exploit the great potential of crowdsourcing to perform large-scale Internet measurement campaigns. We argue that this approach can become an important tool for evaluating innovation solutions, primarily due to the large number of accessible and diverse measurement vantage points. Additionally, we can benefit from the freedom of deploying our own custom-designed measurement tests.

2.2 Experimental setup

We recruit users through the Microworkers crowdsourcing platform to complete measurements on the feasibility of pervasive encryption in the current Internet ecosystem. 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. Another important advantage of using Microworkers for Internet measurements is the possibility to select the MAs 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).

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

Mesurement 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.

Mesurement 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. 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.

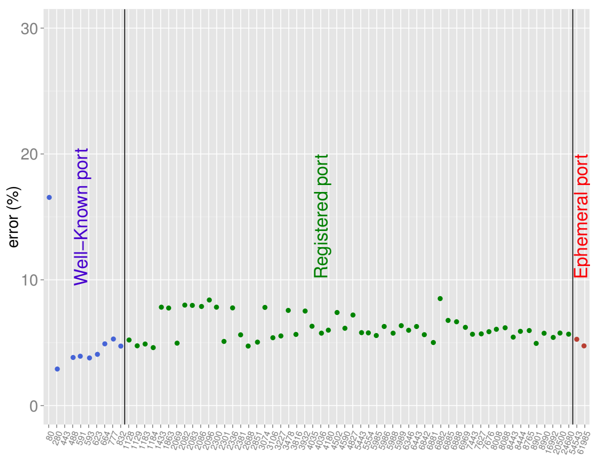


Figure 1: Error rate vs. port number, aggregated results.

3. RESULTS

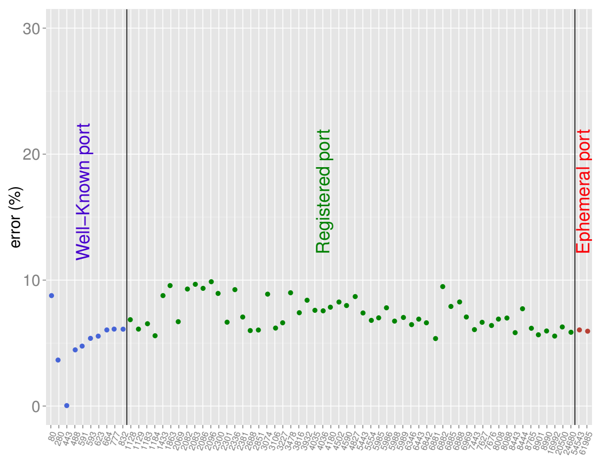
In this section, we present the dataset and we evaluate the performance of HTTP and HTTPS protocols for the tested ports.

3.1 Dataset

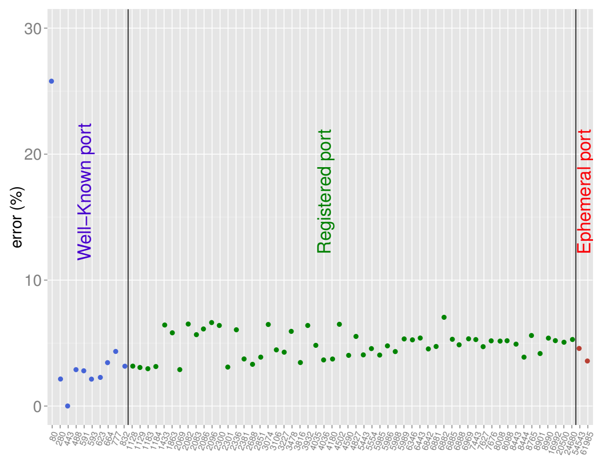
In the campaigns for fixed lines, we recruit 1,165 *workers* from 53 different countries. The MAs are hosted in 286 ASes overall. 79% of the users indicated that they are connecting from *Home*, 10% from a *Company*, 6% from an *University* and 5% from a *Public Hot Spot*. Also, in the case of residential 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 *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 [2]. When processing the data, we

[2]The data set is freely available on *http://it.uc3m.es/amandala/dataset.php*



a) Fixed line



b) Mobile network

Figure 2: Error rate vs. port number.

verify that we only observe data from unique IP addresses, to ensure that the MAs perform the test only once.

3.2 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

$$ERRORS(\%) = (\frac{100}{http}) \times (http - tls) \qquad (1)$$

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

The computed percentage of error for each port is shown in Figure 1. 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.

3.3 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 2 (a) and Figure 2 (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 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.

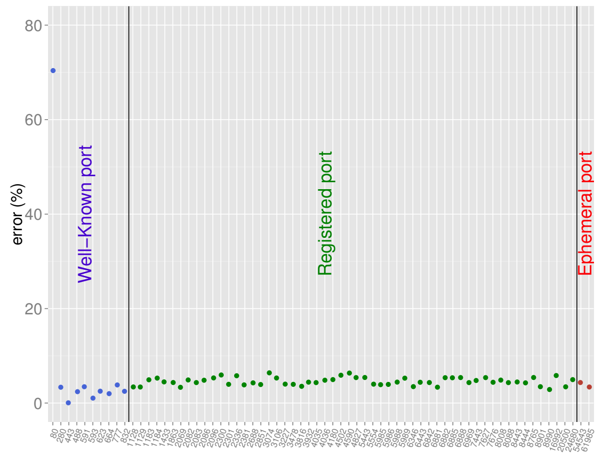
In mobile scenario the average percentage of errors is equal to the 4,54%, considering all ports. However, it is interesting to observe the behavior of middleboxes respect to TLS when port 80 is used. In this case, 25% of the users are not able to perform a TLS connection.

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

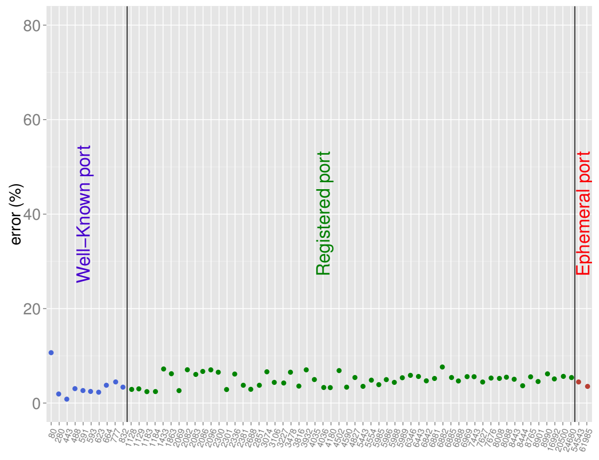
3.4 Proxies

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



a) Mobile proxy



b) Mobile non-proxy

Figure 3: Error rate vs. port number.

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 3 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%.

Table 1: Packets analysis

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

3.5 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.

Table 1 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 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

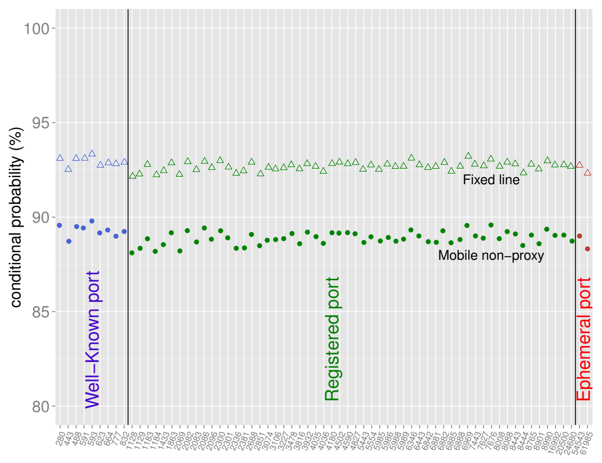


Figure 4: Conditional probability vs. port number.

their policies. This explains why in the case of TLS, we miss a high number of SYN packets.

3.6 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). 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 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. CONCLUSION

In this paper 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. 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 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.

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 such as tcpcrypt [2]. We believe it is a lower bound because there may be middle boxes along the path that do understand (and forward) TLS that would not forward another unknown protocol.

5. ACKNOWLEDGMENTS

We would like to thank Andrew Mcgregor, Bob Briscoe and Rubén Cuevas Rumín for providing helpful information and comments. The work of Anna Maria Mandalari has been funded by the EU FP7 METRICS (607728) project. The work of Marcelo Bagnulo has been funded by the EU FP7 Trilogy2 (317756) project.

6. REFERENCES

[1] https://www.ietf.org/mailman/listinfo/hops. *IETF*, 2015.

[2] A. Bittau et al. The case for ubiquitous transport-level encryption. In *USENIX*, 2010.

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

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

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

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

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

[8] M. Honda et al. Is it still possible to extend TCP? In *ACM IMC*, 2011.

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