Tracking transport-layer evolution with PATHspider

Conference Paper

Author(s):
Trammell, Brian; Kühlewind, Mirja; De Vaere, Piet; Learmonth, Iain R.; Fairhurst, Gorry

Publication date:
2017

Permanent link:
https://doi.org/10.3929/ethz-b-000192306

Originally published in:
https://doi.org/10.1145/3106328.3106336

Tracking transport-layer evolution with *PATHspider*

Brian Trammell, Mirja Kühlewind, Piet De Vaere
ETH Zurich, Networked Systems Group
Switzerland

Iain R. Learmonth and Gorry Fairhurst
University of Aberdeen
Scotland

ABSTRACT

The ossification of the Internet protocol stack, due in large part to mangling of packets by middleboxes, has led to a relatively slow rate of change in today's Internet. We have developed the *PATHspider* active Internet measurement tool which performs one-sided measurements of a variety of transport-layer features and extensions, to investigate these impairments to protocol evolution along an Internet path. Data collected with *PATHspider* can be used both to determine the degree of support for these features, as well as to detect connectivity issues caused by attempting to use them. The wider aim of this effort is to provide quantifiable input to protocol design and deployment choices that can be based on the level of impairment present in the Internet.

This paper details *PATHspider*'s design, and applies it to trace the evolution of deployment of two extensions to TCP, Explicit Congestion Notification (ECN) and the newer TCP Fast Open (TFO); as well as the degree of interference with the Differentiated Services Code Point (DSCP) carried in the IP header. Our ECN results, in particular, expand on a long-term study beginning in 2012, and show continued linear adoption of ECN. Automating *PATHspider* measurements has allowed us to collect far more data than in previous campaigns, allowing us to better distinguish ECN-linked connectivity failures from transient effects. Interestingly, we observe a correlation between ECN-linked connectivity failure in the core of the network with the presence of large-scale, heterogeneous Internet censorship infrastructure.

ACM Reference format:
Brian Trammell, Mirja Kühlewind, Piet De Vaere and Iain R. Learmonth and Gorry Fairhurst. 2017. Tracking transport-layer evolution with *PATHspider*. In *Proceedings of ACM Applied Networking Research Workshop, Prague, Czech Republic, July 2017 (ANRW2017),* 7 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The evolution of the Internet protocol stack is made difficult by a variety of factors: the lack of incentives to be an early adopter of new technology, the difficulty of rolling out new endpoint software to support new protocols and features, assumptions made by devices in the network that tend to impair new features. The result has been an ossification of the protocol stack that leads to a relatively slow rate of change. The protocol engineering community has grown used to this ossification. A common refrain is "we can't do thing X with feature Y because of middleboxes". Seeking to quantify on-path impairments – and to separate features that fail to work for one connection attempt in ten on the Internet from those that do not work on one connection in a hundred million – the authors have developed the *PATHspider* tool [1] to collect information about on-path impairments and endpoint support for new protocols.

When simultaneously used from multiple, topologically diverse vantage points, *PATHspider* can also differentiate impairments close to a server (e.g., at a load balancer or firewall) from the impairments further along the path between that server and the Internet. We infer this path dependency using a tomographic approach: if a protocol feature works from one set of vantage points but not from another, then we expect the impairment exists on the path(s) between the server and those nonworking vantage points. This approach complements more direct measurement of the path with tools such as Tracebox [2], especially because paths that exhibit path-dependent behavior also tend not to pass the ICMP Time Exceeded messages upon which route tracing approaches rely.

PATHspider uses plugins to generate and observe test traffic for each protocol feature it measures. In this paper, we use three of these plugins to examine the current Internet support for two TCP features and one IP feature in various degrees of deployment: Explicit Congestion Notification (ECN) [3], TCP Fast Open (TFO) [4], and Differentiated Services Code Point (DSCP) [5]. We explore this diversity of features both to demonstrate *PATHspider*'s flexibility, as well as to examine different stages and kinds of evolution in the Internet protocol stack.

The ECN results we present are part of a long- term measurement study dating back to 2012 [7, 8]. This period covers a change in the nature of ECN traffic in the Internet: Apple's June 2016 announcement that iOS and macOS devices would begin probabilistic attempts to negotiate ECN by default on the client side[1]. Automating measurements using *PATHspider* has allowed not only a continuation of this measurement campaign, but the collection of much more data, allowing better inferences about path dependency in ECN-linked connectivity failures. We observe that ECN support continues to increase among top web servers, with 73% supporting negotiating ECN when requested. ECN-dependent connectivity failure is limited to 1 in 400 targets.

TFO is a much newer protocol extension than ECN, defined in an experimental RFC [4] only in 2014. Here we see a quite different pattern: only a very few of the top web and DNS servers support TCP Fast Open yet, with the large majority being properties of Google, who proposed TFO and first deployed it. Our sample size is too small to infer any path-dependent interference with TFO, and we also do not see an increase in deployment over the 4 months between two independent measurement campaigns.

ANRW2017, Prague, Czech Republic

DOI: 10.1145/nnnnnnn.nnnnnnn

[1] announced at WWDC 2016, and citing our previous work; see https://developer.apple.com/videos/play/wwdc2016/714/

DiffServ is often used for intra-domain Quality of Service (QoS) signaling but the specification permits remarking of DSCP field in the IP header or dropping of packets according to peering agreements at the borders. However, DiffServ can also be used to provide information from an end host to the network, as recently proposed by WebRTC [9]. Due to intra-domain usage of DSCP network operators often routinely bleach or remark host-provided DSCP. There is current work within the IETF to help to achieve a consistent default DiffServ marking between domains [11]. Our measurements support this work by showing only a small number of partly path-dependent connectivity failures caused by the use of a non-zero DSCP (<1%), however a larger number for bleaching and remarking.

In this study, we examine three different protocol extensions defined after the initial deployment of the protocol, in different stages of deployment, and impaired by different mechanisms of mangling. Server- and client-side defaults for ECN are leading to accelerating deployment, a reduction of impairment, and very rare impairment in the core of the Internet. On the other hand, TCP Fast Open is barely deployed. We see no evidence of network-core or content provider network impairment, though other sources [12] report severe impairment on access networks. DSCP, however, is widely modified in the Internet; DSCP codepoints are often used differently in different networks, and are therefore often bleached on network boundaries.

1.1 Related Work

This study may be seen as a followup to Honda et al's original study of TCP evolvability [6], focusing on a more diverse set of endpoints and on newer protocol features, as well as Bauer et al [13], on the deployment of ECN. In addition to adding data points to this and our own previous work, we revise our estimate of path-dependent impairment in ECN, and therefore the proportion of connection attempts that need more than simple fallback to deal with in-network impairment. More recently, McQuistin et al [14] complemented this data also with measurements of ECN support for UDP traffic. Automating *PATHspider* measurements has allowed us to collect more data, and better isolate transient effects from actual path dependency.

TFO adoption and impairment has been studied in two recent works. First, Christoph Paasch recently presented results at NANOG [12] obtained using the Apple TFO stack. Paasch cited an 80% success rate for TFO connection attempts. The primary impairments observed were blocking of options and blackholing firewalls on access networks. Mandalari et al [15] agree with these findings, using a crowdsourcing platform to run a small study on diverse access networks, finding significant impairments both in negotiation as well as in sending data on the SYN. The present study is the first we are aware of that measures potential impairments to TFO outside access networks, and at TFO adoption among webservers. Our measurements in this work showing limited connectivity impairment from well- connected, unimpaired data center networks corroborate earlier findings that most of these issues are access-network linked.

DSCP modification in the Internet was recently examined by Barik et al [10], who reported results from a small-scale measurement study. It found that the DSCP field was "bleached" to 0 on 45% of measured edge flows, left unmodified on 44%, and otherwise modified on 11%; and that the modification was dependent both on the path, as well as on the codepoint chosen. Our DSCP measurements are largely consistent with those in [10], but examine connectivity dependency, over four orders of magnitude more paths, focusing on the Internet, rather than edge scenarios.

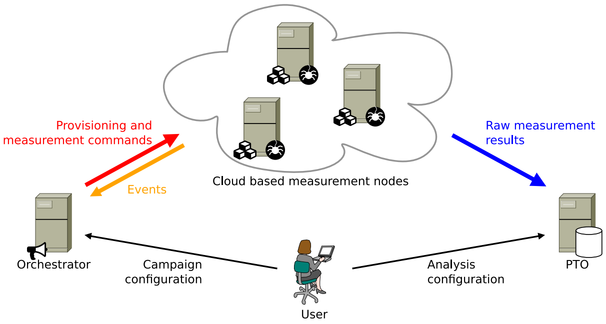

Figure 1: Schematic representations of the complete measurement architecture.

We use *PATHspider*for A/B testing for all three of the protocol feature. Previous measurement tools and campaigns often only focuses on active and passive measurements of TCP [**? ? ?**]. We plan to use *PATHspider*for differential measurements on all layers of the stack and build a database to trace the longitudinal evolution of path transparency of these feature.

2 DESIGN AND IMPLEMENTATION

PATHspider [1] is a measurement tool developed by the authors, as a generalisation of the earlier ECN Spider tool [8] to measure Internet path transparency to various protocol features. It compares connectivity using a vanilla TCP connection between two endpoints – the control traffic – with connectivity using some other protocol or protocol feature – the experimental traffic. *PATHspider* combines active traffic generation with passive observation of the generated traffic to measure not only connectivity dependencies on new features, but also the negotiation of these features, or anomalies in their signaling. The measurements in this paper used *PATHspider* plugins for ECN, TFO, and DSCP testing. These have since been included with *PATHspider* codebase available on GitHub[2].

To enable continuous and automated measurements and analysis using *PATHspider*, we integrated it with the Path Transparency Observatory (PTO) [16] and SaltStack, an open source software project for orchestration of cloud computing in data centers [3]. The operation of the whole system is depicted in figure 1.

The PTO [16] is a data repository, also developed by authors of this paper, for collection of large raw measurement data sets from diverse sources, tools, and measurement campaigns; and for the analysis of this data to derive **observations** on path transparency. An observation is an assertion that a given **condition** was observed on a given path at a given time; e.g. that ECN was successfully negotiated, or that an experimental TFO cookie was seen. The

[2] See https://pathspider.net/ for the *PATHspider* codebase and documentation
[3] https://saltstack.com/

PTO also provides a web front-end[4] for querying and aggregating observation data to answer questions such as "what proportion of observed web servers negotiate ECN"?

SaltStack provides two types of nodes: a single, in our case always-on *master* node, and multiple *minion* nodes running *PATHspider* and performing the actual measurements. The master runs SpiderWeb, a tool that can read campaign configuration files, and uses a time-based job scheduler (e.g., Cron) for periodic measurements. For each measurement run, a new minion is created and connected to the master. Subsequently the master will instruct the minion to fetch the *PATHspider* input file, and to install and execute *PATHspider* itself. When the measurement is completed, *PATHspider* uploads the results to the PTO and the master will request to the cloud provider to destroy the minion measurement node.

Because the orchestrator will create and configure all measurement nodes, there are only two elements of the measurement infrastructure the user has to interact with: the orchestrator (running the SaltStack master), and the PTO (to configure analysers).

2.1 Explicit Congestion Notification

ECN is a TCP extension that is negotiated in the TCP handshake and allows the sender to mark its packets as ECN-capable transport (ECT) to indicate support to the routers on the network path to signal congestion other than by packet drop. ECN has been standardized in 2001 and after that been widely implemented in all main operating systems. However, due to initial deployment problems, there was little immediate deployment, and ECN was often blocked or stripped by firewalls in response to these issues. Since our first measurement in 2012, server-side support has increased continuously, mainly due to OS updates supporting the ECN server-mode, where ECN is only negotiated if requested by the client. Besides measuring negotiation support (of servers), *PATHspider* also measures connectivity success for ECN negotiation, as those that hindered initial deployment, or when attempting to use ECN if negotiated successfully. The measurements are the information needed to turn on ECN negotiation by default on the client as currently under investigation by Apple.

Our measurements leads to the following conditions that can be observed: **ECN negotiation succeeded**; **ECN negotiation failed** but the TCP connection was established; **ECN connectivity works** for both connection attempts; **ECN connectivity is broken** but connectivity without ECN negotiation attempt was established; **ECN connectivity is transient** as the TCP connection with ECN negotiation was established, but not without[5]; or **complete connection failure**.

Note that transient connectivity may cause both "broken" and "transient" conditions to be observed. We use a generalization of the methodology in [8] to correct for this "noise floor", as well as to determine whether or not connectivity dependency is **site-dependent**, impaired by the target server itself or a device on all paths to the server (e.g., a load balancer); or **path-dependent**, impaired by devices only present on some paths to the server. We do this by running *PATHspider* on seven vantage points at once, for path diversity. And to reduce transient noise, we run three trials in parallel for each target on each vantage point.

In section 3.1, a target is considered to have site-dependent ECN connectivity issues only if, across all trials and vantage points, only broken ECN connectivity is observed. Targets with site-dependent connectivity are reachable by clients who negotiate ECN and implement a fallback on lost SYN; the current revisions of all major client operating systems now do this after the acceptance of our kernel patch [8] into the Linux upstream. A target is considered to have path-dependent ECN connectivity issues only if two vantage points disagree on ECN connectivity across all trials, and no other vantage point shows transient or offline behavior.

2.2 TCP Fast Open

TFO is a rather new, experimental protocol extension that enables transmission of payload data in the TCP SYN packet and therefore reduces latency by one Round Trip Time (RTT) [4]. In an initial TCP connection between a client and server, a client can request a cookie using an option in the SYN packet that can be used on subsequent connections to the same server to already attach payload data in the SYN packet. The TFO plugin for *PATHspider* effectively has three instead of two phases. For each target it first connects with "vanilla" TCP without any extensions or options, it then connects to the server using TFO twice: first to request a cookie, then to determine whether data on SYN using the cookie is ACKed. It can test both HTTP and DNS over TCP using TFO. For HTTP, it issues a simple `GET / HTTP/1.1` request for the host associated with the name. For DNS, it issues an A query for the name of the DNS server.

This test methodology allows us to classify connection attempts with TFO as follows, in increasing order of brokenness: **TFO works** (cookie received, data on SYN ACKed); **TFO data not ACKed** (cookie received, only SYN ACKed); **TFO data failure** (cookie received, SYN with data fails); **TFO not negotiated**; **TFO connection failure** (RST or drop with TFO option); or **complete connection failure**.

2.3 Differentiated Services

DiffServ provides a 6-bit code point in the Differentiated Services (DS) field in the IP header for packet classification. The value of this code point is used to signal to routers on the network path how to treat the packet when forwarding, with common classifications being default treatment (best effort, value 0) Best Effort (BE), Assured Forwarding (AF) and Expedited Forwarding (EF). The DSCP plugin for *PATHspider*, for the baseline case, creates a TCP connection to a webserver using a DSCP value of 0 and, for the experimental case, a second TCP connection with a DSCP value of 46 for EF.

The plugin observes the completion of the 3-way handshake and the DSCP value on packets which are returned. With this test methodology we can observe the following conditions: **DSCP connectivity works** for both connections; or **DSCP connectivity broken** when the the EF code point is used. We also observe the individual codepoints received for every flow, and retain this raw data.

[4] currently in alpha testing; a public access URL will appear in the camera-ready, and is available from the authors upon request.

[5] Since there is no reasonable model for a host that *only* responds to ECN-enable TCP handshakes at this time, we clearly consider this case as a transient measurement error.

Table 1: ECN summary statistics for popular Web servers, June 2016 and January 2017

June 2016				January 2017				
IPv4		IPv6		IPv4		IPv6		Description
n =617 873		n =24 472		n =675 289		n =90 531		
hosts	pct	hosts	pct	hosts	pct	hosts	pct	
9221	1.49%	2637	10.78%	12583	1.863%	3621	4.000%	Completely failed to connect
432544	68.78%	20262	76.77%	498866	73.874%	82722	95.232%	Capable of negotiating ECN, of which:
11718	1.86%	2167	8.21%	15000	3.007%	6622	8.005%	Never mark ECT
-	-	-	-	30	0.006%	16	0.019%	Mark ECT1
1112	0.18%	964	3.65%	1851	0.274%	23	0.025%	Failed to connect w/ECN

3 RESULTS

All measurements in this study were performed against a list of targets built from the Alexa top 1 million websites, the MAMI Public Targets List (PTL)[6] and/or the OpenDNS top million domains.

3.1 Explicit Congestion Notification

PATHspider was run on two separate occasions to continue our longitudinal measurement study that began in 2012 [7]: once in June 2016, and a larger measurement campaign beginning in December 2016 and January 2017. Connectivity and negotiation statistics from a single trial for each at a single vantage point (the Digital Ocean Amsterdam data center) are summarized in table 1.

Negotiation numbers show a continuation of the linear trend of servers capable of ECN negotiation, as well as a spike in both ECN support and general stability of the IPv6 web. In January, we note about one in a thousand servers marking ECT1, which may have some impact on recent efforts within the IETF to reclaim the ECT1 codepoint from the defunct ECN Nonce experiment [17] for low-latency services [18, 19].

Applying our multiple-trials, multiple-vantage methodology described in section 2.1 to reduce the effect of transient connectivity on measuring ECN site and path dependency, we find 1034 (0.153%) IPv4 and 4 (0.042%) IPv6 addresses where connectivity is always dependent on ECN negotiation (site dependent), and 194 (0.029%) IPv4 and 2 (0.021%) IPv6 addresses where connectivity dependence on ECN negotiation is dependent on path. This is less than a quarter of the connectivity risk we measured in [8], which we account to two possible causes. First, since those measurements were taken, Apple has made ECN negotiation default on the client side for a proportion of all connections from macOS and iOS; this provides an incentive for content providers to ensure that ECN negotiation doesn't cause increased latency for their content. Second, our current methodology was designed to better reject false positives due to transient connection failure than the previous one, so we provide a tighter upper bound.

ECN linked connectivity impairment is dominated by a small number of networks in a few countries: 73% of servers with site-dependent connectivity impairment, and 58% of servers with path dependency, are either in mainland China or South Korea. As seen in Figures 2 impairment affects more than one in a five hundred Chinese and South Korean servers, but this impairment is not temporally stable. For comparison, we see path-dependent connectivity failures for less than 0.01% of all servers in the US where the US is the country with the largest number of target points in our data set. We have also observed either path- or side-dependency for nearly all servers in North Korea, however, our sample set is too small to draw a meaningful conclusion with only about a dozen servers for this country. We note that this behavior is consistent with the inconsistency previously reported in both the Chinese [20] and South Korean [21] Internet censorship systems. A primary remaining cause of ECN impairment in the Internet is therefore intentional if collateral damage from Internet censorship, caused by devices that are *designed* to interfere with TCP sessions.

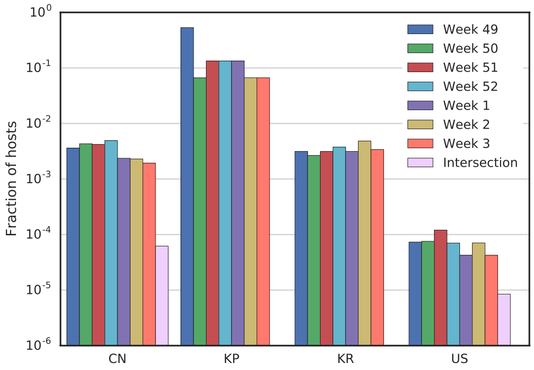

Figure 2: Proportion of sites per country with path dependency over six weeks in December 2016/January 2017

3.2 TCP Fast Open

We did two measurements of TFO support on the Alexa top million websites, an initial measurement in October 2016, and a follow-up measurement in January 2017. The initial study ran from six vantage points hosted by Digital Ocean in Amsterdam, Frankfurt, London, San Francisco, Singapore and Toronto on 12-13 October 2016. As the sample size of TFO-supporting websites is too small to discern any path-dependent behavior, the January 2017 followup ran from a single site in Frankfurt 16 January 2017, to determine if there had been any discernible change in TFO deployment in

[6] see https://github.com/mami-project/targets; the PTL was created in the aftermath of Alexa's December 2016 announcement that access to the top million websites list would be made subject to payment terms; the later announcement that Cisco would open the OpenDNS list for the same purpose has made this homegrown project less necessary.

Table 2: TFO summary statistics for popular Web servers and associated DNS servers, October 2016 / January 2017

Web, Oct ’16		Web, Jan ’17		DNS, Jan ’17		
n =673 230		*n* =939 680		*n* =53 267		
hosts	pct	hosts	pct	hosts	pct	description
18 777	2.79%	29 839	3.18%	4 906	9.21%	Completely failed to connect
211	0.031%	177	0.019%	26	0.049%	Failed to connect w/TFO option
653 681	97.1%	908 464	96.7%	48 276	90.6%	Did not negotiate TFO
578	**0.086%**	**866**	**0.092%**	**56**	**0.105%**	Negotiated TFO (exchanged a cookie); of which:
563	97.4%	830	95.8%	54	96.4%	ACKed data on SYN †
0	0%	0	0%	2	3.57%	Failed connection with data on SYN
16	2.77%	33	3.81%	0	0%	Returned a cookie on ACKed data
11	1.90%	12	1.39%	2	3.57%	Responded with a 6-byte cookie
15	2.60%	31	3.58%	0	0%	Responded with an experimental option †
485	83.9%	690	79.7%	53	94.6%	are in AS15169 (Google)

the intervening three months, and to additionally measure TFO support of popular authoritative DNS servers.

Our October Web targets were taken from the Alexa top million websites list retrieved in early October 2016. Our January web and DNS targets were taken from the MAMI PTL which resolves all addresses for each domain, resulting in the larger target set for January 2017 that includes a union of the set of domains used in two previous ECN measurement studies. The DNS targets are the set of NS results listed for the domains used to generate the list, and reflect authoritative servers. Our results are summarized in table 2.

The deployment of TFO on popular Web servers seems to be dominated by Google. It is supported by less than one web server in a thousand, showing little uptake by other servers. This is also unsurprising, given that it requires both kernel- and application-level changes both on clients and servers, making the process behind the adoption curve more complex than that driving Explicit Congestion Notification (ECN) adoption in webservers. We see no evidence of a change in TFO support on web servers during the period reported in this study.

Given that TFO has recently been promoted as a first step toward using DNS over TLS over TCP for DNS query confidentiality[7], we decided to measure TFO support on popular authoritative DNS servers as well. Here, Google is even more dominant. We see evidence of only one non-Google server correctly negotiating TFO.

That said, a group of 12 servers, all affiliated with a Spanish insurance company, show an interesting cluster of anomalies, indicating an implementation of TFO unique to this enterprise, which does not interoperate with the TFO implementation the Linux kernel on the Debian 7 images we used for testing. These sites negotiate TFO using option number 34, use a 6-byte (as opposed to 8-byte) TFO cookie, return a cookie on the ACK regardless of whether one was sent, and do not ACK data on the SYN.

We also found 31 unrelated servers that negotiated TFO using the older experimental option (number 254), indicating they are running older TFO implementations[8].

We examined the 546 sites negotiating TFO in October 2016 more closely, attempting to connect on three separate runs from six vantage points. Of these sites, none showed stable path dependency on TFO negotiation, but 16 did show some transient instability in TFO negotiation, indicating either changes in the TFO configuration state of the server, or load-balancing to servers with different TFO configurations.

We saw no evidence of connection failure linked to data on the SYN after TFO is negotiated. Of the 211 hosts that had apparent connectivity problems with the TFO option set in October 2016, only four hosts (all IPv4) had consistent connectivity dependency from all vantage points, indicating that the server, or a middlebox close to the server, may be blocking packets with the TFO option set. We do note that no study without control of both endpoints can distinguish servers that do not support TFO from those where TCP options are stripped.

3.3 Differentiated Services

The DSCP plugin was used from seven vantage points hosted by Digital Ocean in Amsterdam, Frankfurt, London, New York, San Francisco, Singapore and Toronto on the 30th September 2016 and on the 30th January 2017. The second run also included a Digital Ocean hosted vantage point in Bangalore. Connections were attempted to 673,230 IP addresses from each of these vantage points, with the same list of targets for both measurement runs. Our results are summarized in table 3.

For the targets where the baseline – outgoing DSCP 0 (default) – connection succeeded, we found that 99.95% of targets also succeeded with the experimental – outgoing DSCP 46 (EF) – connections. For both IPv4 and IPv6, in both measurement runs, we observed that <0.05% of targets would fail to complete a TCP handshake for the experimental flow from all vantage points.

However, in this <0.05%, we saw targets where a connection using the EF code point never succeeded from any vantage point. 30 of these targets exist in the network of a US-based co-location provider, and were observed via 284 unique paths in the September measurement run to fail if the EF code point is set. The remainder were distributed across other ASNs, many of which are located in China, again indicating a cross-connection with traffic manipulation due to censorship and a heterogeneous infrastructure deployment.

[7] see https://portal.sinodun.com/wiki/display/TDNS/TCP+Fast+Open and the IETF DPRIVE working group

[8] †: Since TFO negotiation fallback works on Linux by attempting to use TFO over option 254 only after no cookie is received with option 24, and *PATHspider* only uses two TFO connections we did not test any of these sites to see if they would ACK data on the SYN; these two rows in the table are mutually exclusive

Table 3: DSCP summary statistics for popular Web servers, September 2016 and January 2017

September 2016				January 2017				
IPv4		IPv6		IPv4		IPv6		Description
n =620 611		n =52 766		n =620 611		n =52 766		
hosts	pct	hosts	pct	hosts	pct	hosts	pct	
35 768	5.76%	24 422	46.28%	63 177	10.18%	28 985	54.93%	Completely failed to connect
584 843	**94.24%**	**28 344**	**53.72%**	**557 434**	**89.82%**	**23 781**	**45.06%**	Successfully connted with DSCP 0 (default); of which:
2 321	0.40%	156	0.55%	1 770	0.32%	124	0.52%	Failed to connect when DSCP 46 (EF) used;
2 170	0.37%	154	0.54%	1 334	0.24%	121	0.50%	but succeeded from at least one vantage point
584 692	**99.99%**	**28 342**	**99.99%**	**556 998**	**99.92%**	**23 778**	**99.98%**	Successfully connected with 46 (EF)

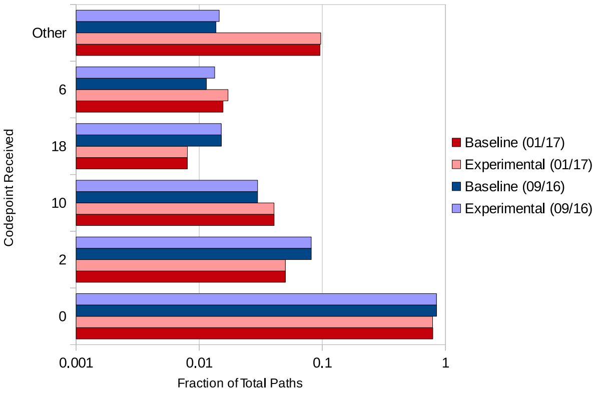

Figure 3: DSCP codepoints received for popular Web servers, September 2016 and January 2017

We further investigated the code points that were received on the incoming packets from the targets during the measurements. In general, applications will decide on the code point to use when sending data, and it is not expected that this decision would be influenced by codepoints received previously in the same connection set by the communication partner (or network). First of all, we observed that we received a default treatment (BE) code point on both the baseline and experimental flows for 81.47% distinct paths in September 2016 and 78.84% in January 2017.

We observed that for paths that connected fine for both flows, the code point received in replies would change between the baseline and experimental flow on 0.96% of paths for September 2016 and on 1.13% in January 2017. More than half of the targets for these paths showed that this behaviour was path-dependent indicating that the codepoints may be being rewritten in the core of the Internet, at peerings between networks, and not in the local or target networks.

In total, we observed 53 distinct code points in both September 2016 and January 2017 received between the vantage points in the measurement. A summary of the top code points received can be found in figure 3. It is interesting to note that the 3 most observed code points in both measurements, after the default, are all private use not registered by standards action within the IETF. This may indicate that only the first 3 bits of the field have been rewritten to zero. For the case of 2, it is possible that the original code point may have been 18 (AF21) or 34 (AF41). The GSM Association has made recommendations to use AF21 for web traffic [22], and so we believe it is likely that this was the case though cannot prove this.

For 0.08% of paths for which both connections succeeded in September 2016 and 0.09% for the paths in January 2017, we observed that the code point that had been sent was the DSCP reflected back. Approximately one-fifth of the targets for these paths showed this behaviour from all successful vantage points with the remainder being path dependent, again indicating code point rewriting in the core of the Internet.

4 CONCLUSIONS AND FUTURE WORK

We have presented three large-scale Internet measurement studies done with *PATHspider*, examining deployment and deployability of technologies at multiple layers of the stack. Not surprisingly, different forces lead to different impairments at each layer. DSCP is mainly impaired by widespread network operations practice. TFO is largely unimpaired within the Internet core, but the deployment of TFO-capable servers lags, in part due to widespread access network impairment of TCP options [12]. However, the adoption of TFO may be superseded by the deployment of QUIC [23], which uses UDP encapsulation [24] to thwart on-path modifications to TCP. ECN, on the other hand, is further in its deployment, and client- and server-side default configuration of ECN provides an incentive to remove core- and edge-network impairments to using ECN. We note that the majority of remaining impact on connectivity linked to ECN negotiation is correlated with large-scale national Internet censorship; we imply that the manipulation of TCP sessions done by this rather heterogeneous infrastructure is interfering with ECN negotiation and signaling.

Analysis and detailed intermediate results for the results presented in this paper are available[9]. Raw data from this study is stored in the PTO, and access is currently available upon request.

There are two major areas of ongoing work. First, the authors are presently adding mobile vantage points through the MONROE project[10]. Second, *PATHspider* remains under active development[11] including plugins for additional extensions, such as MPTCP and enhancements such as the integration of simultaneous route tracing and packet mangling analysis inspired by Tracebox [2], in order to provide additional information about the path at the time of measurement.

[9] TFO data is available at https://github.com/mami-project/must-go-faster, and ECN data at https://github.com/mami-project/ecn-conspiracy

[10] https://monroe-project.eu

[11] see https://pathspider.net

5 ACKNOWLEDGEMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 688421, and was supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number 15.0268. The opinions expressed and arguments employed reflect only the authors' views. The European Commission is not responsible for any use that may be made of that information. Further, the opinions expressed and arguments employed herein do not necessarily reflect the official views of the Swiss Government. Many thanks to Andreas Germann for his work on an initial TFO survey in June 2016.

REFERENCES

[1] Learmonth, I., Trammell, B., Kühlewind, M., Fairhurst, G.: PATHspider: A tool for active measurement of path transparency. In: First ACM/IRTF Applied Networking Research Workshop, Berlin, Germany (Jul 2016)

[2] Detal, G., Hesmans, B., Bonaventure, O., Vanaubel, Y., Donnet, B.: Revealing Middlebox Interference with Tracebox. In: Proceedings of the 2013 Conference on Internet Measurement Conference. IMC '13, Barcelona, Spain, ACM (2013) 1–8

[3] Ramakrishnan, K., Floyd, S., Black, D.: The addition of explicit congestion notification (ecn) to ip. RFC 3168, RFC Editor (September 2001) http://www.rfc-editor.org/rfc/rfc3168.txt.

[4] Cheng, Y., Chu, J., Radhakrishnan, S., Jain, A.: Tcp fast open. RFC 7413, RFC Editor (December 2014) http://www.rfc-editor.org/rfc/rfc7413.txt.

[5] Nichols, K., Blake, S., Baker, F., Black, D.L.: Definition of the differentiated services field (ds field) in the ipv4 and ipv6 headers. RFC 2474, RFC Editor (December 1998) http://www.rfc-editor.org/rfc/rfc2474.txt.

[6] Honda, M., Nishida, Y., Raiciu, C., Greenhalgh, A., Handley, M., Tokuda, H.: Is It Still Possible to Extend TCP? In: Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference. IMC '11, Berlin, Germany, ACM (2011) 181–194

[7] Kühlewind, M., Neuner, S., Trammell, B.: On the state of ECN and TCP options on the Internet. In: Passive and Active Measurement Conference, Hong Kong, China (2013) 135–144

[8] Trammell, B., Kühlewind, M., Boppart, D., Learmonth, I., Fairhurst, G., Scheffenegger, R.: Enabling internet-wide deployment of explicit congestion notification. In: Passive and Active Measurement Conference, Brooklyn, USA (2015) 193–205

[9] Jones, P., Dhesikan, S., Jennings, C., Druta, D.: Dscp and other packet markings for webrtc qos. Internet-Draft draft-ietf-tsvwg-rtcweb-qos-14, IETF Secretariat (March 2016)

[10] Barik, R., Welzl, M., Elmokashfi, A.: How to say that youâĂŹre special: Can we use bits in the IPv4 header? In: First ACM/IRTF Applied Networking Research Workshop, Berlin, Germany (Jul 2016)

[11] Geib, R., Black, D.: Diffserv-interconnection classes and practice. Internet-Draft draft-ietf-tsvwg-diffserv-intercon-06, IETF Secretariat (June 2016) http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-diffserv-intercon-06.txt.

[12] Paasch, C.: Network support for TCP Fast Open. Presentation at NANOG 67 (January 2016)

[13] Bauer, S., Beverly, R., Berger, A.: Measuring the state of ecn readiness in servers, clients,and routers. In: Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference. IMC '11, Berlin, Germany, ACM (2011)

[14] McQuistin, S., Perkins, C.S.: Is explicit congestion notification usable with udp? In: Proceedings of the 2015 ACM Conference on Internet Measurement Conference. IMC '15, New York, NY, USA, ACM (2015) 63–69

[15] Mandalari, A.M., Bagnulo, M., Lutu, A.: TCP Fast Open: initial measurements. In: Proceedings of the ACM CoNext 2015 Student Workshop, Heidelberg, Germany (December 2015)

[16] Neuhaus, S., Edeline, K., Donnet, B., Gubser, E.: Towards an observatory for network transparency research. In: Proceedings of the Applied Networking Research Workshop (ANRW'16), ACM (2016)

[17] Spring, N., Wetherall, D., Ely, D.: Robust explicit congestion notification (ecn) signaling with nonces. RFC 3540, RFC Editor (June 2003)

[18] Black, D.: Explicit congestion notification (ecn) experimentation. Internet-Draft draft-ietf-tsvwg-ecn-experimentation-00, IETF Secretariat (December 2016) http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-ecn-experimentation-00.txt.

[19] Schepper, K.D., Briscoe, B., Tsang, I.: Identifying modified explicit congestion notification (ecn) semantics for ultra-low queuing delay. Internet-Draft draft-briscoe-tsvwg-ecn-l4s-id-02, IETF Secretariat (October 2016) http://www.ietf.org/internet-drafts/draft-briscoe-tsvwg-ecn-l4s-id-02.txt.

[20] Ensafi, R., Winter, P., Mueen, A., Crandall, J.R.: Analyzing the Great Firewall of China Over Space and Time. In: Proceedings on Privacy Enhancing Technologies 2015, Philadelphia, USA (June 2015)

[21] Kim, P.: Studying the Internet Censorship in South Korea. A Slice of Kimchi - IT Security Blog (October 2016)

[22] Association, G.: Guidelines for ipx provider networks version 12.0. (2016) http://www.gsma.com/newsroom/wp-content/uploads/IR.34-v12.0.pdf.

[23] Hamilton, R., Iyengar, J., Swett, I., A.Wilk: QUIC: A UDP-Based Multiplexed and Secure Transport. Internet-Draft draft-hamilton-quic-transport-protocol-00, IETF (July 2016)

[24] Trammell, B.: Can we run the Internet over UDP? Measurement and Analysis of Protocols Research Group, IETF 95 (April 2016)