Public Review for

Tracking the deployment of TLS 1.3 on the Web: A story of experimentation and centralization

Ralph Holz, Jens Hiller, Johanna Amann, Abbas Razaghpanah, Thomas Jost, Narseo Vallina-Rodriguez, Oliver Hohlfeld

TLS 1.3 has finally arrived after a long period of development, and it has the promise of being faster and more secure. TLS 1.3 is now being rapidly adopted by browser vendors and rolled out by a number of providers, dominantly CloudFlare. In this paper, the authors use a large number of vantage points to study the adoption of TLS 1.3 by a large number of domains and endpoints, and the central role played by a few big players. The authors discuss the phenomenon of centralization and how big industry players lead the development of web protocols that the rest of the Internet will then have to adopt.

The paper provides a good historical and technical overview of the new TLS protocol and its adoption by various domains and providers, as well as comparison with the previous versions. One of the aspects that the reviewers liked was that the paper considers both mobile and web datasets, and provides an analysis of both ecosystems. The paper and its findings should be of interest to service providers, operators, and Internet measurement enthusiasts.

Public review written by
Hamed Haddadi
Imperial College London & Brave Software

Tracking the deployment of TLS 1.3 on the Web: A story of experimentation and centralization

Ralph Holz[1,2] , Jens Hiller[3] , Johanna Amann[2,4] , Abbas Razaghpanah[4] , Thomas Jost[3] , Narseo Vallina-Rodriguez[4,5] , Oliver Hohlfeld[6]
[1]University of Twente, [2]University of Sydney, [3]RWTH Aachen University, [4]ICSI,
[5]IMDEA Networks, [6]Brandenburg University of Technology
r.holz@utwente.nl,{hiller,jost}@comsys.rwth-aachen.de,{johanna,abbas}@icir.org,narseo.vallina@imdea.org,hohlfeld@b-tu.de

ABSTRACT

Transport Layer Security (TLS) 1.3 is a redesign of the Web's most important security protocol. It was standardized in August 2018 after a four year-long, unprecedented design process involving many cryptographers and industry stakeholders. We use the rare opportunity to track deployment, uptake, and use of a new mission-critical security protocol from the early design phase until well over a year after standardization. For a profound view, we combine and analyze data from active domain scans, passive monitoring of large networks, and a crowd-sourcing effort on Android devices. In contrast to TLS 1.2, where adoption took more than five years and was prompted by severe attacks on previous versions, TLS 1.3 is deployed surprisingly speedily and without security concerns calling for it. Just 15 months after standardization, it is used in about 20% of connections we observe. Deployment on popular domains is at 30% and at about 10% across the *com/net/org* top-level domains (TLDs). We show that the development and fast deployment of TLS 1.3 is best understood as a story of experimentation and centralization. Very few giant, global actors drive the development. We show that Cloudflare alone brings deployment to sizable numbers and describe how actors like Facebook and Google use their control over both client and server endpoints to experiment with the protocol and ultimately deploy it at scale. This story cannot be captured by a single dataset alone, highlighting the need for multi-perspective studies on Internet evolution.

CCS CONCEPTS

• **Security and privacy** → **Security protocols**; • **Networks** → **Network measurement**.

KEYWORDS

TLS, HTTPS, active scanning, passive monitoring, Android

1 INTRODUCTION

Over the last decade, the Web's most important security protocol, Transport Layer Security (TLS), has come under increasing scrutiny, and a long list of vulnerabilities and flaws have been addressed [38]. TLS 1.3 is the latest version of TLS. The new protocol version is almost a complete redesign, with striking differences to previous versions in the protocol flow and the use of cryptography.

Unlike previous versions, the TLS 1.3 design, development, and deployment phases saw an unprecedented involvement by important industry players whose services depend on a well-performing and secure TLS, in particular Google, Facebook, and Cloudflare. They provided input to the protocol design as well as telemetry data revealing incompatibilities with defective implementations of prior TLS versions.

Here, we report on our measurement of the deployment of TLS 1.3 from a very early stage. Given the distinctive design and development processes of the new protocol, this is one of the rare chances for the network and security communities where large-scale data is available to track experimentation and adoption of a new protocol and understand how operators react to its promise but also to potential issues.

To study the deployment and use of TLS 1.3 in breadth and depth, we collect and analyze more data from more vantage points than any previous study. Our passive measurements from monitoring network connections in the high billions capture the entire period from the conception of TLS 1.3 until today. Our active scans begin in 2017-10, nearly a year before the final IETF RFC, when the message format had mostly stabilized, all major features had been added, and industry players had begun their TLS 1.3 tests. We track the deployment until the end of 2019, more than a year after the standardization process ended. We cover close to 275×10^6 domains, including 54 country-code top-level domains (ccTLDs). We also analyze the use of TLS 1.3 on Android devices, where our data reaches back to the very early drafts (2015-11). Using a single data source—as done in many studies—is insufficient to capture the evolution of a protocol like TLS 1.3, calling for future studies to incorporate multiple perspectives.

Our primary contribution is a comprehensive portray of the evolution of TLS 1.3 deployment, its use, and the impressive way in which deployment happens faster than for any previous TLS standard and possibly any new security protocol. We show that the activities of a small number of cloud providers is a constant, dominating factor during both the design and deployment phases. Specifically, we group our contributions under the following two themes:

1) Experimentation with TLS 1.3. The new protocol enjoyed considerable support even at draft stage. About ten percent of the domains on the Alexa Top1M list enabled it by late 2017, a year before standardization, with deployment on the wider Internet at around 2%. This is also when our passive monitoring, including of the mobile ecosystem, identifies a small percentage of TLS 1.3 connections. The experimental nature of this support is reflected in timely and rapid update

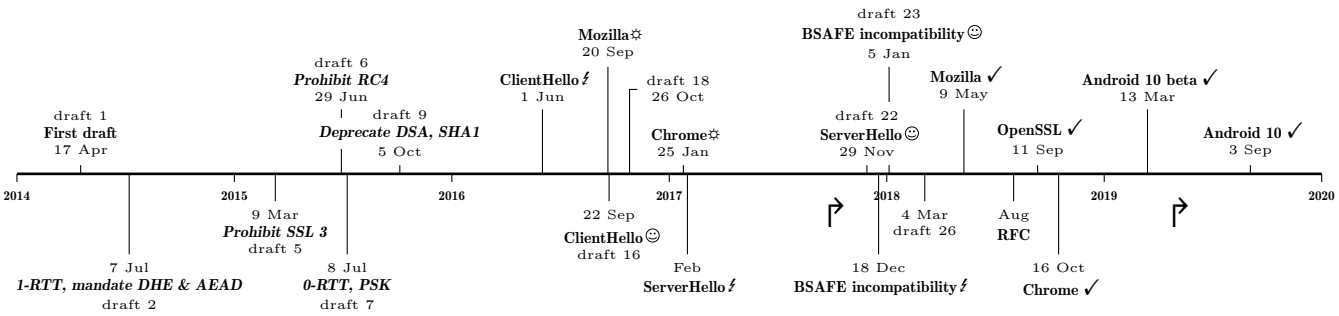


Figure 1: Timeline for TLS 1.3; important changes in italic (see [50]). We highlight some critical issues (↯) found in trials and fixed (☺). Support was sometimes first optional (☼), later a default (✓). The arrows (↱) show the start times of draft/RFC scans.

cycles following each new draft. The actors behind this early support and experimentation are Cloudflare and Facebook on server-side, although we also find support in the networks of some Virtual Private Server (VPS) hosters. On client-side, we describe the trials run by Google and especially Facebook, who use customized versions of TLS 1.3.

2) TLS 1.3 on the centralized Internet. Following standardization, TLS 1.3 is deployed much faster than historical experience would have suggested. We trace this to the strong centralization that the Internet is experiencing, with an enormous number of domains hosted or front-ended by just a few very large providers. We demonstrate this for domains across all our domain sets. Investigating support among eleven of the largest cloud and frontend providers, we find that support is far from uniform. Cloudflare, Facebook, and (partially) Google are early movers, while other giants such as Amazon, Azure, or Alibaba lack support for TLS 1.3 even by late 2019.

Online resources. We wish to support other researchers aiming to replicate or reproduce our results. We give details in Appendix B. The following site links to datasets, code, and an extended technical report accompanying this paper:

https://tls13.globalsecuritylabs.org

2 BACKGROUND

The IETF standardized TLS 1.3 in 2018-08 [49] as a reaction to a growing list of weaknesses in previous versions pertaining to flaws in the protocol flow and the use of weak cryptographic algorithms [5, 8, 12, 38, 40]. The new protocol is backward-compatible only in the sense that client and server may still agree to negotiate an older version. The logic of the protocol handshake has significantly improved: encryption is used as early as possible, even for server certificates and many extension headers. Payloads are generally transmitted after just one round-trip (1-RTT) and the previous fragility of session resumption has been addressed with a new unified mechanism that combines several methods and allows to send payload data in the first message (0-RTT), albeit at the price of losing replay protection [42]. Forward Secrecy is now mandatory, and only Authenticated Encryption with Associated Data (AEAD) ciphers are allowed. Together, these changes result in a protocol with much better performance and higher security guarantees. We give an overview of the most relevant changes that TLS 1.3 introduces in Table 1.

Table 1: Main differences between TLS 1.3 and earlier versions.

	< TLS 1.3	TLS 1.3
Payload after...	2 RTTs	1 RTT (optional 0-RTT)
Resumption	Tickets	Tickets, pre-shared keys, 0-RTT, forward secrecy
Encryption after...	Handshake	2nd handshake message
Static RSA	Allowed	No (→ forward secrecy)
Static Diffie-Hellman	Allowed	No (→ forward secrecy)
Non-AEAD ciphers	Allowed	No (→ crypto resilience)
Forward Compatibility	N/A	Prepared (GREASE [19])

Figure 1 shows a timeline of the draft process; it also shows when software support became available in various important implementations. Stakeholders began supporting and experimenting with TLS 1.3 variants long before standardization had concluded. Cloudflare was the first large provider to enable TLS 1.3 for its customers as early as 2016-09 [41]. On the client side, Mozilla's Firefox and Google's Chrome were the first browsers to include support for TLS 1.3 in their codebases in 2016-09 (Firefox 49) and 2017-01 (Chrome 56), respectively. Both ran small beta programs in 2017-02. Facebook also experimented with TLS 1.3 drafts in their apps and infrastructure [31, 43]. By the time of standardization (2018-08), they reported that more than 50% of their global Internet traffic was already served via TLS 1.3 [31]. Android added support in the Android 10 beta in 2019-03 and made it the default in the official release in 2019-09 [22]. Most Android apps use OS defaults to set up TLS connections [45] and will profit from the support without modification.

However, the upgrade path is not without obstacles. Incompatibilities arose while TLS 1.3 was under development: trials by Google showed that some servers and middleboxes on the Internet reacted to the new version number, used in the *ClientHello* and *ServerHello* messages, by dropping the traffic entirely [15, 25, 47, 48]. Since draft 16, TLS 1.3 always sets the version field in the *ClientHello* to TLS 1.2. The *actually* supported versions are sent in a new extension. Since draft 22, the same concept is also used in the *ServerHello*.

Table 2: Overview of our datasets and (primary) use of each.

Name	Type	Vantage Point	Start	End	Interval	Volume	Used to investigate ...
ActDraft	active	Aachen, DE	2017-10	2019-05	day/week	147-172 $\times 10^6$ domains	trials: drafts on servers
ActRFC	active	Sydney, AU	2019-05	2019-11	monthly	273-277 $\times 10^6$ domains	server support: final version
Passmon	passive	N. America	2012-02	2019-11	continuous	$>400 \times 10^9$ connections	use by clients & servers
Android	passive	Global	2015-11	2019-11	continuous	11.8 $\times 10^6$ connections	use in Android ecosystem

This makes TLS 1.3 the first TLS version that explicitly works around defective implementations.

To boost forward compatibility for future protocol changes, RFC 8701 (GREASE [19]) uses currently reserved values liberally to ensure no implementation will implement the standard only partially and terminate connections with unknown versions, extensions, or cryptography. However, TLS 1.3 drafts also needed to break compatibility between each other, as with draft 23, which fixed an issue due to an invalid implementation choice in a widely used library (BSAFE, [17]).

3 DATASETS AND METHODOLOGY

We choose our data sources to inspect the burgeoning TLS 1.3 ecosystem from various important angles. We use *(i)* active Internet scans to identify TLS 1.3 support and characterize deployment; *(ii)* passive monitoring to study TLS connections and understand the usage of TLS 1.3 in practice; and *(iii)* application traffic captured on mobile devices through crowd-sourcing to gain insights into the use of TLS in the mobile ecosystem. We discuss the ethical considerations of our data collections in Appendix A. The appendix also contains information about repeatability and reproducibility. Table 2 summarizes our data collection.

Active scans. We create scan targets from domain lists and zone files. We combine three public top lists (Alexa Top1M, Majestic, Umbrella) with domains from the zonefiles for the *com/net/org* top-level domains (TLDs). This yields between 164-166 $\times 10^6$ domains for each scan. We add domains from the zonefiles of the new generic top-level domains (ngTLDs), which adds between 21-23 $\times 10^6$ domains. Finally, we acquire 88 $\times 10^6$ domains in 54 country-code top-level domains (ccTLDs) from *ViewDNS*, which are collected in web crawls.[1]

We choose these *domain sets* to determine whether TLS 1.3 deployment is different between them. Following Scheitle *et al.* [54], one should expect the Alexa Top1M list to be a *'solid choice of functional websites frequently visited by users'*, and *com/net/org* TLDs as suitable to *'obtain a reasonably general picture of the Internet'*. Domains in the ngTLD set are known to be used for squatting and preemptive registrations [33]. To the best of our knowledge, our study is the first to specifically consider ccTLD domains. We would expect differences between countries, depending on the market shares of the most important cloud providers and hosters in each.

We scanned the Internet from two locations. From RWTH Aachen in Germany, we scanned the Alexa Top1M list daily and most zonefiles weekly, utilizing a modified version of *zgrab* [4]. The scans began in 2017-10, about 10 months before standardization was complete. We implemented support for new drafts when they became available, although we observed major CDNs (especially Cloudflare) to be occasionally faster. In such cases, we are unable to determine the precise moment when the respective provider enabled the new version. We use the version extension to announce several draft versions. If servers choose a version we have not implemented yet, we abort the handshake gracefully and store the *ServerHello*.

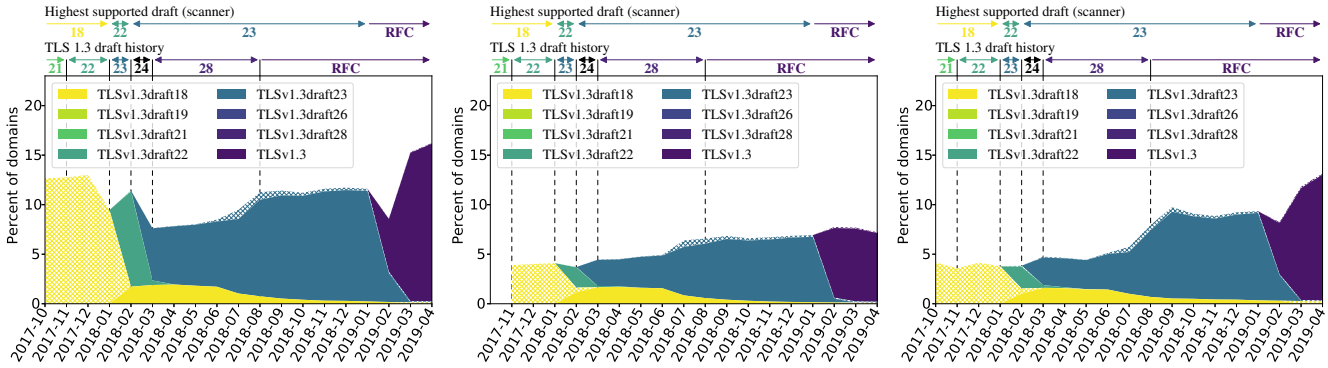
From 2019-05, monthly scans of the standardized version take place from the University of Sydney in Australia[2]. The choice of interval is based on hardware availability. We resolve domains to A records using *massdns* [1] and use *zmap* [30] to identify IP addresses with open port TCP/443. Independent of vantage point, we find the port open on about 90% of domains on the Alexa Top1M list, on about half the *com/net/org* and ccTLDs domains, and on about 40% of ngTLD domains. We use the scanner introduced in Amann *et al.* [9], using the TLS 1.3 implementation of Go.

Passive monitoring. We use data from the ICSI Notary [11], a large-scale observation effort of TLS with a set of monitors in North America. Since its inception in 2012, the Notary has observed more than 400 $\times 10^9$ TLS connections using the Zeek network monitor [3]. Typically, between three to eight sites contribute data at any given time. Data is collected in operational environments. This means we cannot quickly redistribute new monitoring code. The data collection is also a best effort: outages, packet drops, and misconfigurations are rare but possible. Our data can be expected to have some bias: we monitor only research networks, and the contributing sites are all located in North America. Given the huge volume across our sites, however, the aggregate data should capture many properties of real-world TLS traffic.

Android data. We inspect data gathered by the Lumen Privacy Monitor [45], a privacy-enhancing research tool for Android [46], to understand the trial and deployment phases of TLS 1.3 on this mobile platform. Lumen analyzes mobile traffic in user space to help users stay on top of their mobile traffic and privacy by reporting network flows and dissemination of personal data, with an option to block undesired traffic. Lumen correlates traffic flows with app identifiers and process IDs to match TLS flows to the processes that generated them. At present, more than 22,000 users from over 100 countries have installed Lumen. The dataset contains accurate, anonymized traffic fingerprints for more than

[1] For reasons unknown to us, ViewDNS (viewdns.info) does not include domains in *.uk*.

[2] We did not scan in 2019-09 due to maintenance work.



(a) Alexa Top1M **(b) com/net/org** **(c) All zones without com/net/org**

Figure 2: Early experimentation, view in active scans. Atop, we list the highest draft we advertise, below this we add a timeline of draft releases. Solid areas refer to scans with full handshakes; hatched areas to scans with incomplete handshakes (see Section 3).

92,000 Android apps. It excludes mobile browsers to preserve user anonymity (see Appendix A). For this paper, we analyze 11.8×10^6 TLS connections from 56,221 apps, across multiple versions, connecting to 149,389 domains.

Identifying centralized services. To understand the impact of Internet centralization on TLS 1.3 deployment, we choose eleven globally acting cloud, frontend, and hosting providers that either disclose their IP ranges or where the IP ranges are readily identified on *bgp.he.net*. Cloudflare offers Web frontending services; Amazon AWS, Azure, and DigitalOcean focus on on-demand cloud computing, including VPSes. We add Facebook, Google, and Alibaba as prominent cloud service and hosting providers and Squarespace and GoDaddy as important web hosters. Finally, we add OVH as a 'classic' hosting provider where one can rent VPSes. In our early measurements, we identify SingleHop as a VPS provider with an interesting deployment pattern and add them as well. SingleHop is US-based but also has international customers. Note that we scan *domain names from zonefiles*, *i.e.,* we are unlikely to hit many targets that belong to provider core infrastructure. We use the domain categorization service provided by VirusTotal to determine whether certain types of services are more likely to deploy TLS 1.3.

Limitations. Server-side logs from large providers were not available to us. Such data may add an interesting perspective in future investigations, as would data from the iOS platform. We do not scan IPv6 due to lack of support in the university network. Go implements only three of the five ciphers defined in the RFC. We do not support Secure Sockets Layer (SSL), the precursor to TLS deprecated in 2015.

4 EARLY EXPERIMENTATION

Experiments with TLS 1.3 draft versions took place from early on. We find two consistent patterns across all our data sets: rapid update cycles and activities constrained to very few but prominent actors.

4.1 Rapid, timely update cycles

Figure 2 shows the evolution of deployment of TLS 1.3 draft versions across various DNS zones. We obtain a monthly view by taking the union of all scan results from the respective month. We advertise draft 18 from 2017-10 on, although we do not complete the full handshake yet. As can be seen in Figure 2a (hatched area), deployment was already surprisingly high on domains on the Alexa Top1M list. Twelve percent of domains select draft 18 as their preference when offered by our scanner. The number is considerably lower for domains outside the Alexa Top1M list (Figures 2b and 2c).

We support drafts 18-22 in full from 2018-02 onwards. By this time, most servers already report their support as well and complete the handshake successfully, largely for draft 22. Almost ten percent of domains support draft 22, and about 1.7% select draft 18. Our passive observations from the ICSI Notary and Lumen paint a very similar picture. In the Notary data, depicted in Figure 3, first, barely noticeable support for TLS 1.3 is advertised by servers at roughly the same time our scanners also pick up on it, *i.e.,* by 2017-10. In the Lumen data covering the Android platform, shown in Figure 4, we observe an increase of TLS 1.3 usage with a delay of just a few weeks. The timely implementation and deployment of new draft versions is apparent throughout our observation period. Figure 2 shows how new drafts replaced previous ones extremely quickly. However, we also note that some servers continue to use draft 18 surprisingly long. We find support well into 2018, when the middlebox problems relating to *ClientHello* and *ServerHello* messages had been resolved (draft 22, 2017-11). We identify the same rapid update and deployment cycle in the mobile ecosystem (Figure 4).

Before 2017-11, neither Notary nor Lumen data show any real use of TLS 1.3. The first noticeable increase is a slight bump to 7% just after 2017-11 (Figure 3) which coincides with a test deployment of draft 22 in Chrome 63 between 2017-12-05 and 2017-12-19 [17]. This draft still suffered from an incompatibility issue, which was resolved in 2018-01. With support for the new draft 23, client support rises sharply

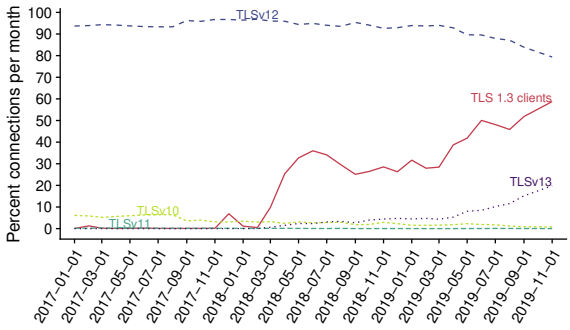


Figure 3: TLS 1.3 connections, view by Notary. Servers select TLS versions from client offers; clients offering TLS 1.3 in red.

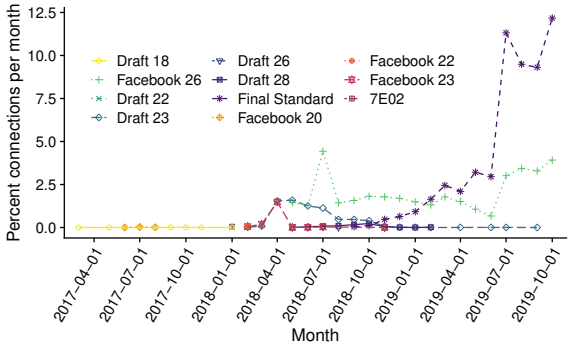


Figure 4: TLS 1.3 versions negotiated as fraction of all TLS connections in Android, view by Lumen.

(Figure 3). When the RFC is published in 2018-08, the continued climb coincides with the final RFC quickly displacing all draft versions. On Android, we find the fraction of TLS 1.3 connections rise from 0.01% of all TLS connections in 2018-01 to 16% in 2019-10 (Figure 4).

In our active scans (Figure 2), overall deployment rises very slowly until 2019-02. Technical limitations between 2018-01 and 2019-02 prevented us from tracking the precise, subsequent switches from draft 23 to the final RFC. However, most servers have moved to the final RFC by 2019-03. The first apparent drop in support from 2018-02 to 2018-03 among Alexa Top1M domains is actually due to Alexa changing their domain ranking algorithm [54]. The second apparent and temporary drop in support for the standardized version is an artefact caused by our scanner not immediately supporting the new *Hello Retry Request* feature.

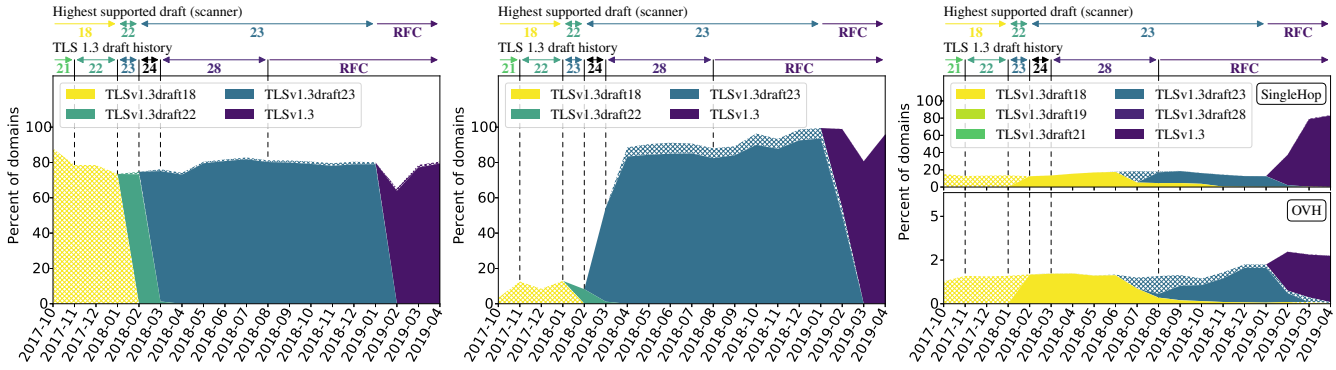
4.2 Actors driving deployment

We observe only three, large actors to be the primary drivers behind early deployment and experimentation: Cloudflare, Facebook, and Google, although the latter conducted early tests only for short periods of time [16, 17] and uptake is slower. In much smaller numbers, we also find deployment on VPS providers. Some of our chosen providers, in particular Microsoft, Amazon, and Alibaba do not enable TLS 1.3 at any significant scale even by late 2019. Figure 5 summarizes the results from our active scans, broken down for different operators. We identify different activity patterns.

Cloudflare. Cloudflare made TLS 1.3 a default for customers of their Free and Pro tiers in 2016-09 [41]. Customers in higher tiers could opt in. By 2017-10, Cloudflare already accounts for 75% of all TLS 1.3-enabled domains in our active measurements. The provider moves rapidly from draft to draft, with very timely tracking of each new version. Support is enabled on most domains operated by Cloudflare (Figure 5a). In 2018-05, when Mozilla made TLS 1.3 the default and Chrome enabled optional support, Cloudflare enabled TLS 1.3 as a default for customers of higher tiers [7]. However, we find only a very modest increase of six percentage points in our scans. Either most customers did not belong to the higher tiers or they switched off TLS 1.3 again.

Facebook. When looking only at active scans, one might believe that Facebook did not experiment much with official drafts before draft 23, which then sees a rapid deployment phase, reaching approximately 83% on Facebook's public domains by early 2018 (Figure 5b). Interestingly, however, this picture changes nearly completely when we consider the mobile ecosystem. The Facebook family of apps (Facebook, Messenger, Instagram) implement and support custom versions from early 2017 on, with clearly differentiated, only slightly overlapping phases. Facebook used their control over both apps and infrastructure to experiment extensively with their custom versions (Facebook 20, 22, 23, 26). Facebook informed us that the custom versions are identical to the draft versions, except for the code point for the version. Using custom versions allowed Facebook to fully deploy each draft for their apps while avoiding a need for roll-backs on server side in case of an incompatibility with other implementations. Our data confirms that Facebook's servers select the Facebook-specific version when communicating with the app; in scans, the web servers choose the normal draft versions. Only drafts 23 and 26 find significant use by Facebook, most others barely show up in our Lumen data. Draft 26 replaces draft 23 nearly instantly and completely around 2018-05. Perhaps more surprisingly, our Lumen data suggests that Facebook apps continue to use the custom version 26 at least until the end of our observation period. According to the RFC changelog [50], there is no difference between draft 26 and the RFC version as far as the actual protocol is concerned. Hence this has no negative operational consequences.

Google. One might assume that Google, one of the companies pushing for TLS 1.3 deployment and the creator of Android and Chrome, would also have reason to experiment with TLS 1.3 at scale and from early on. Indeed, Google already used its control over both servers and Chrome for *short* tests of draft 18 (2016-02; Chrome beta) [16] and draft 22 (2017-12; Chrome stable) [17] before we scanned for the respective version. In terms of *steady support*, however, we can



(a) Cloudflare (b) Facebook (c) SingleHop and OVH

Figure 5: Early adopters, view in active scans. Atop, we list the highest version we advertise, with a timeline of draft releases. Solid areas refer to complete handshakes, hatched areas to incomplete ones (see Section 3). Note the scales of the y-axes.

identify only a tiny bump, comfortably below 0.5%, of draft 23 deployment from about 2018-04 until 2018-07 in our active scans. Even after the release of the RFC, we cannot identify significant deployment until the first quarter of 2019, when it reaches nearly 5%. On Android, we find no use of TLS 1.3 by Google apps before 2018, around the time draft 23 was published. However, we note that Chrome, like all browsers, is not among the apps we monitor. Nevertheless, Google's TLS 1.3 support became manifest with the release of Android 10's beta in 2019-03, which includes native support.

Other infrastructures. We find some support for TLS 1.3 drafts on VPS providers. The deployment patterns, however, show little correlation to major milestones in the development of TLS 1.3. For OVH, we find support for draft 18 at around 1.7% of domains hosted in their IP space (Figure 5c). This lasts until 2018-06, when this draft version is replaced with draft 23 within two months. However, even though the standard was finalized in 2018-08 and server support was added to OpenSSL in 2018-09, we do not find a rapid uptake in TLS 1.3 usage in general. Draft 23 remains in use until 2019-02, when it is rapidly replaced by the RFC version, now on 2.9% of domains. A provider that turns up on our radar rather unexpectedly is SingleHop who is based in the US and, like OVH, also a globally acting operator. Remarkably, SingleHop's deployment pattern is nearly identical to OVH's. Their deployment is significantly larger than OVH's, however, around 15% until the beginning of 2019, when RFC deployment speeds up and reaches nearly 85% within just four months. The clearly delineated, short, and coinciding transition periods of OVH and SingleHop are more indicative of provider actions than of singular customers making changes. It is not clear what caused the providers to switch between drafts, nor why they kept the draft version in use for much longer than necessary. The middlebox incompatibilities do not coincide with the transition periods, and the release history of both OpenSSL and Google's variant, BoringSSL, show no correlation to the transition periods.

On other VPS providers and hosters, we find barely any engagement with draft versions at all. For domains on Squarespace, for example, we find no support before 2019-02. Then, however, deployment is rapid: nearly 90% of scanned domains on Squarespace use TLS 1.3 by 2019-03.

5 TLS 1.3 ON A CENTRALIZED INTERNET

The phase of early experimentation with TLS 1.3 is followed by a sustained deployment effort that proceeds at a much faster pace than for TLS 1.2. Security issues are not the reason; instead, the centralization of Internet services on platforms of very few providers (Cloudflare and Google in particular) is the dominant factor for this fast deployment of TLS 1.3 across our domain sets. However, other major operators such as Amazon and Azure show almost no uptake more than a year after standardization. On a per-country basis, deployment of TLS 1.3 in the ccTLDs is also linked to *regional* centralization: it occurs as a consequence of country-wide customer preferences for smaller operators. Surprisingly, some important economies lag massively behind in deployment.

Adoption pattern: TLS 1.3 vs. TLS 1.2. The adoption of TLS 1.3 is significantly faster than that for TLS 1.2. Standardized in 2008, TLS 1.2 does not appear in significant numbers in Notary data before mid-2013, and usage does not reach 50% until 2014. The dominant TLS library on UNIX-like systems, OpenSSL, supported it from 2012-03 [2], Chrome from 2013-08, and Mozilla from 2014-04. The late, but then fast uptake of TLS 1.2 deployment was considerably accelerated by the publication of attacks on RC4 and later the Lucky13 attack [38]. In contrast, TLS 1.3 is already negotiated in 19.5% of connections 15 months after standardization, and nearly 60% of clients advertise support for it (Figure 3). There is no severe vulnerability that would motivate operators to switch away from TLS 1.2: instead, we identify the centralization of the Internet as the most dominant factor for the fast introduction of TLS 1.3. The commitment by large providers is likely motivated by the improved performance and security promised by TLS 1.3 [31, 41]. A further, critical

Table 3: Share of all TLS 1.3/TLS-enabled domains, connections, and IPs across chosen providers and Facebook, as of 2019-05.

	ActRFC								Passmon			
	% Alexa Top1M		% *com/net/org*		% ngTLDs		% ccTLDs		% connections		% IPs	
TLS →	1.3	1.x	1.3	1.x	1.3	1.x	1.3	1.x	1.3	1.x	1.3	1.x
Cloudflare	**59.8 (1)**	**13.5 (1)**	**35.4 (1)**	**4.8 (2)**	**70.6 (1)**	**17.3 (1)**	**32.7 (1)**	**3.4 (2)**	**6.9 (3)**	1.3 (5)	**66.9 (1)**	**5.1 (2)**
Google	**11.3 (2)**	**5.7 (3)**	1.4 (4)	2.9 (6)	0.6 (5)	2.6 (5)	0.6 (6)	2.1 (5)	**50.0 (1)**	**22.0 (2)**	**7.3 (2)**	**1.6 (3)**
Squarespace	**4.8 (3)**	1.0 (8)	**29.8 (2)**	3.6 (4)	**7.1 (2)**	1.7 (6)	**5.5 (2)**	0.6 (6)	<0.1 (7)	<0.1(11)	<0.1(10)	<0.1(11)
Amazon	0.8 (4)	**7.5 (2)**	0.6 (6)	**4.3 (3)**	**0.7 (3)**	**7.6 (2)**	0.5 (7)	**3.3 (3)**	0.3 (4)	**28.6 (1)**	1.8 (4)	**63.4 (1)**
SingleHop	0.7 (5)	0.4 (9)	**2.0 (3)**	0.5 (8)	0.3 (7)	0.2(10)	0.6 (5)	0.1 (9)	<0.1 (8)	<0.1(10)	0.9 (6)	0.1 (9)
OVH	0.7 (6)	3.8 (4)	1.0 (4)	3.5 (5)	0.7 (4)	3.9 (4)	**1.0 (3)**	**5.8 (1)**	<0.1 (9)	0.2 (7)	0.3 (7)	0.5 (5)
DigitalOcean	0.6 (7)	1.7 (6)	0.5 (7)	0.9 (7)	0.3 (6)	1.3 (7)	0.6 (4)	0.6 (7)	<0.1 (6)	0.2 (6)	1.2 (5)	0.7 (4)
Azure	0.1 (8)	1.3 (7)	<0.1 (9)	0.4 (9)	<0.1(10)	0.3 (8)	<0.1 (8)	0.3 (8)	<0.1(10)	**8.6 (3)**	0.1 (8)	0.5 (6)
Facebook	<0.1 (9)	<0.1(11)	<0.1 (8)	<0.1(11)	<0.1 (9)	<0.1(11)	<0.1 (9)	<0.1(11)	**26.8 (2)**	3.0 (4)	**3.0 (3)**	0.2 (8)
GoDaddy	<0.1(10)	2.8 (5)	<0.1(11)	**15.2 (1)**	<0.1(11)	**6.4 (3)**	<0.1(11)	2.7 (4)	<0.1(11)	<0.1 (9)	<0.1(11)	0.4 (7)
Alibaba	<0.1(11)	0.1(10)	<0.1(10)	0.1(10)	<0.1 (8)	0.3 (9)	<0.1(10)	<0.1(10)	<0.1 (5)	0.1 (8)	0.1 (9)	<0.1(10)

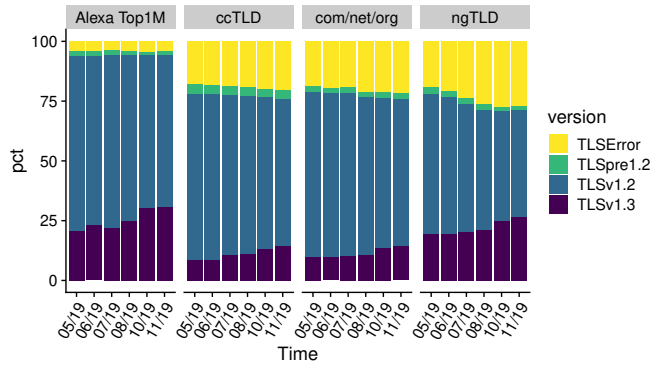


Figure 6: Deployment since 2019-05, view from active scans. Percentages based on domains with open port 443.

difference to TLS 1.2 is the forward compatibility enabled by GREASE [18]. This mechanism is in active use: 56.7% of the connections we see at the Notary contain at least one GREASE marker. A wide use of this mechanism should make changes and new designs of future TLS versions considerably easier.

Adoption pattern: steady growth. Figure 6 visualizes data from our active scans of the RFC version of TLS 1.3, starting from 2019-05, grouped by domain set, *i.e.,* Alexa Top1M domains, ccTLDs, the *com/net/org* TLDs, and the ngTLDs (also see Section 3). Deployment rises steadily, at a relatively fast pace. In 2019-05, deployment on Alexa Top1M-listed domains has reached about 21%, up six percentage points from 2019-04. By 2019-11, it is 31%. The numbers are considerably lower in the ccTLD group and *com/net/org*, where they grow from 7.5% to 11.3% and 5.8% to 9.3%, respectively. In the ngTLD group, we find a deployment that is as strong as on Alexa Top1M domains.

Deployment across providers and domain sets. Table 3 (left) summarizes the results of our scan in 2019-05, differentiating by domain set. Table 3 (right) breaks down the traffic monitored by the Notary, distinguishing between percentage of connections and percentage of destination IP addresses.

Cloudflare is well-known for its large market share, and it is no surprise that the company's share among TLS-enabled domains is generally the largest or second-largest in any domain set. Our scans identify Cloudflare for 13.5% of Alexa Top1M domains and more than 17% of domains in the ngTLDs. Table 3 shows that our other chosen providers are generally responsible for (significantly) fewer domains. Google and Amazon still have a sizeable share among Alexa Top1M domains, and GoDaddy and Squarespace among domains in *com/net/org*.

The actors we identified during the early experimentation phase remain active and dominant. In fact, the strong deployment of TLS 1.3 on domains on the Alexa Top1M list and domains in the *com/net/org* TLDs and ngTLDs is due to Cloudflare enabling it: the company is responsible for 59.8%, 35.4%, and 70.6% of TLS 1.3-enabled domains, respectively. We note that many domains in the ngTLDs are known to be preemptively registered [33]. It is quite plausible that Cloudflare's share here is due to the company offering a free DNS and reverse-proxy service. Interestingly, as Table 3 shows, Cloudflare does by far not receive most TLS 1.3 connections: their share is below seven percent. Google operates a respectable number of 11.3% TLS 1.3-enabled domains on the Alexa Top1M list. Not all Google-hosted domains are already TLS 1.3-enabled, however: the number is 40% in 2019-05 but grows to nearly 60% by 2019-11.

We also find newcomers that were not active during the experimentation phase. A surprisingly strong factor for domains in *com/net/org*, the ngTLDs, and the ccTLDs is Squarespace: around 30%, 7%, and 6% of TLS 1.3-enabled domains, respectively, are hosted by this company, despite the lower share among TLS-enabled domains in general. However, these domains are not particularly often accessed, as Table 3 shows.

The giant providers Amazon, Azure, and Alibaba, who were not active during the early phase, have not moved much since the RFC, either. They are responsible for fewer TLS-enabled domains in general on the one hand; on the other hand, Amazon and Azure receive sizeable volumes of TLS traffic[3] (Table 3). Inspecting our scan data for Alexa Top1M domains, we find only very moderate increases in deployment,

[3]While traffic to Alibaba is low from the Notary's vantage point, we note that traffic in China is very likely to be much larger.

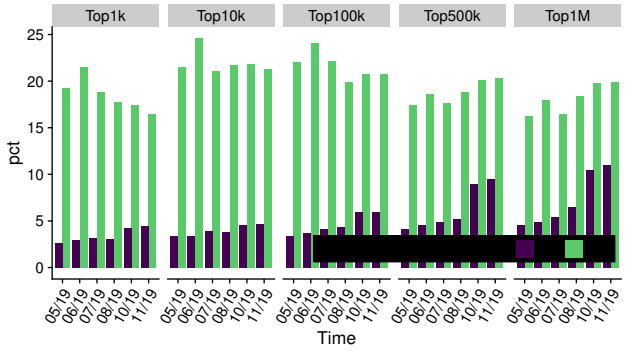


Figure 7: Evolution of deployment in the Alexa Top1M. Section 3 explains our choices concerning cloud providers.

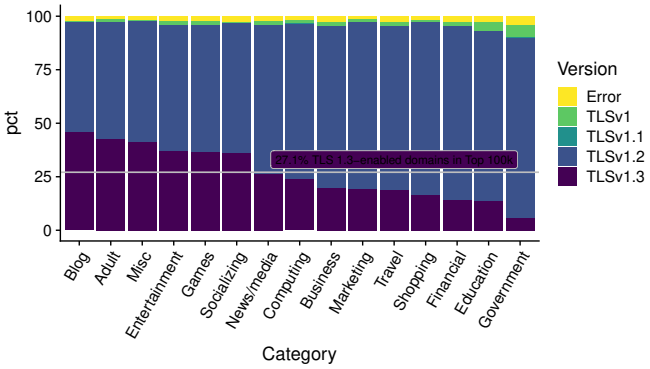


Figure 8: Top 15 categories, Alexa Top100k of 2019-07.

even when checking again in 2019-11. Alibaba's deployment on their domains rises from nearly zero to about 3.5%, which is similar to Amazon (around 4%). In the case of Azure, deployment does not even reach 2% by 2019-11.

We find more movement among the smaller VPS providers. Until the second half of 2019, the numbers for DigitalOcean and OVH remain around 8% and 4% (OVH), respectively, but then rise to nearly 20% in both cases, possibly because customers are updating gradually. Although the company follows a similar business model, SingleHop IPs show a deployment around 80% by 2019-04 already. We contacted SingleHop to inquire whether they used a form of front-end for users (similar to Cloudflare) but did not receive a reply.

Unlike Squarespace, GoDaddy does not enable TLS 1.3, which is surprising given their large customer base. Support is basically non-existent in 2019-05 and remains so in 2019-11.

Popular domains and industries. Figure 7 shows the deployment of TLS 1.3 across Alexa ranks, distinguishing between domains hosted on our chosen providers and those hosted via other means. Surprisingly, TLS 1.3 deployment in the Alexa Top1k domains is initially rising, but then falling again in our provider ranges. We see the same rise, followed by a drop and stagnation in the Alexa Top10k and Alexa Top100k. This is largely due to a drop in the percentage of TLS 1.3-enabled domains that Cloudflare manages.

To understand whether particular industries drive adoption, we use VirusTotal to classify domains by purpose and resolve the domains on the Alexa Top100k. Figure 8 shows the results for the 15 most common domain categories in 2019-07. Note that 27.1% of all Alexa Top100k domains deploy TLS 1.3. We find above-average deployment mostly in categories that can be classified as various forms of (social) entertainment, ranging from blogs to adult sites. We find by far the lowest deployment on government domains, but education and finance domains also have low deployment. Organizations in these categories are known to be more compliance-driven, which may explain a slower uptake. Perhaps somewhat surprisingly, we also find sites for online shopping to have a low adoption. A possible explanation would be operational concerns: customers can be easily lost if the upgrade to a new protocol causes problems with browsers. To answer this question, one would need to know the distribution of browser versions among the respective sites' customers (also see Section 7).

Providers in ccTLDs. Table 4 shows the deployment of TLS 1.3 across 54 ccTLDs as a percentage of all TLS connections as of 2019-08. We find a large disparity among countries, from deployment of 80% down to the low single-digit percentages. Checking against the scan from 2019-05, deployment is increasing across all ccTLDs (also see Figure 6). Surprisingly, some ccTLDs make enormous jumps: Denmark's fraction doubles to more than 50%, and Sweden's fraction increases from just 4.5% to 38.6%. At the low end, Germany moves from 3.8% in 2019-05 to 8.3%. However, some strong economies like France and Japan have TLDs where deployment stays well under 5% and does not improve significantly.

Due to Cloudflare's prominent role, we test the correlation between the percentage of TLS 1.3-enabled domains (pct_T) and percentage of domains in Cloudflare ranges (pct_C). A scatter plot shows six outliers; we have $s^2(pct_T) = 270.0$. Hence, we use Kendall's tau, which yields $\tau = 0.30$, indicating at least a weak correlation between pct_T and pct_C ($p = 1 \times 10^{-3}$). The outliers—*cf* and *tk* on the one hand, and *dk*, *ua*, *sk*, and *se* on the other—have particular characteristics, as we explain below. If we eliminate them, the variance drops to $s^2(pct_T) = 34.2$. Using Pearson's method, the correlation coefficient is $r = 0.56$ ($p = 36.6 \times 10^{-6}$), *i.e.,* a relatively strong correlation.

The deployment at the high end is easy to explain. The registries of the Central African Republic (*cf*) and of Tokelau (*tk*) allow the registration of domain names free of charge. The 87,500 domains in *cf* resolve to just over 53,000 distinct IP addresses; in *tk*, the rougly 153,000 domains resolve to about 68,000 IPs. Cloudflare is hosting most domains: about 46,000 in the case of *cf* and 48,000 for *tk*. It is plausible that Cloudflare's free tier of services is also attractive to registrations in these TLDs.

Table 4: TLS 1.3-enabled ccTLD domains (2019-08, as fraction of TLS-enabled domains). *xn–p1ai* is punycode for the Russian Federation's TLD.

Rank	TLD	%	Rank	TLD	%	Rank	TLD	%	Rank	TLD	%	Rank	TLD	%	Rank	TLD	%
1	cf	80.0	11	us	20.1	21	cc	15.1	31	cz	12.4	41	es	8.3	51	hu	4.8
2	tk	76.3	12	co	18.8	22	il	15.0	32	sg	11.8	42	it	7.9	52	fr	4.4
3	dk	53.2	13	au	18.5	23	ro	14.7	33	tv	11.8	43	kz	7.8	53	za	3.7
4	ua	42.9	14	ru	17.9	24	at	14.2	34	nl	11.3	44	in	7.7	54	jp	3.3
5	sk	41.5	15	su	17.1	25	eu	13.9	35	ch	11.3	45	pe	7.6			
6	se	38.6	16	no	16.8	26	xn--p1ai	13.5	36	ie	10.5	46	cl	6.6			
7	pl	29.2	17	la	16.6	27	nz	13.1	37	ca	10.1	47	mx	6.4			
8	me	24.7	18	ir	16.4	28	my	12.9	38	tw	8.9	48	rs	5.9			
9	io	24.5	19	cn	16.2	29	tr	12.7	39	br	8.7	49	ar	5.4			
10	ma	21.5	20	be	16.1	30	gr	12.5	40	de	8.3	50	pt	5.4			

We investigate several TLDs with high and relatively low deployment, *i.e.,* deployment around 40% or better vs. single-digit deployment. For the ccTLDs on ranks 3–6—Denmark, Ukraine, Slovakia, and Sweden—we find a consistent pattern. By far the most domains are *not* hosted by Cloudflare or any of our chosen providers: between 90.6% (*dk*) and 96.2% (*sk*) are not front-ended by any of them. To understand the hosting situation better, we resolve the Domain Name System (DNS) nameserver records for TLS-enabled domains. We find that regionally focused hosters contribute most to the higher numbers for TLS 1.3. For example, *one.com* is active in both Sweden and Denmark, hosting about a third of all domains. It is responsible for the majority of TLS 1.3-enabled domains (86% in Sweden's case). The cases for Ukraine and Slovakia are similar, with two providers, respectively one, making for about 75% of TLS 1.3-enabled domains.

This is contrasted by the hosting situation in ccTLDs with little TLS 1.3 deployment. For the bottom five in Table 4, we find a much wider spread of hosters. In all countries, Cloudflare is actually in the first or second place in terms of total number of front-ended domains with TLS support. With the exception of Japan, where *value-domain.com* is responsible for more than 70% of TLS 1.3 domains, the company is also responsible for most TLS 1.3 domains. However, Cloudflare's *overall* share of domains is always below 4%.

6 RELATED WORK

Previous academic efforts have characterized and studied different aspects of TLS and the X.509 Public Key Infrastructure (PKI), including the general state of the ecosystem and certificate validation [6, 10, 24, 30, 35], certificate revocation [9, 56, 57], vulnerability discovery [5, 8, 12, 40], Certificate Transparency [23, 32, 51, 53, 55], and TLS/HTTPS support [34, 52]. Regarding TLS 1.3, several papers examined the protocol and proposed improvements and new features [13, 27, 39], cryptographic schemes [14], or discovered vulnerabilities and flaws through protocol verification and cryptographic analysis [20, 21, 26, 29, 36, 37].

Only a very limited number of studies have measured TLS 1.3 deployment and support. None provide a comprehensive analysis of TLS 1.3; they usually gained anecdotal insights about ongoing deployment efforts of TLS 1.3 as a by-product of their attempts to answer more general research questions about TLS use and deployment in the wild. In previous work of some of our authors, Kotzias *et al.* [38] perform a longitudinal analysis of TLS deployment over five years. The authors focus on changes in TLS deployment and industry practices caused by the disclosure of protocol vulnerabilities. The authors include only a brief report on TLS 1.3 deployment as of April 2018 (used in 1% of observed connections). The passive monitoring data that Kotzias *et al.* use is the same as ours (the Notary) and hence our measurement results are the same until April 2018; however, TLS 1.3 deployment started in earnest after that date, and our observation period extends to the end of 2019. A 2017 study analyzing Lumen data (which we also use in our paper) reported marginal support of TLS 1.3 extensions [45].

7 DISCUSSION AND CONCLUSION

We presented a longitudinal study of deployment and use of TLS 1.3, from the early experimentation stages to deployment after standardization. A key finding is the high deployment that TLS 1.3 already enjoys, which shows that TLS 1.3 is adopted much faster than previous TLS versions. We identify two major reasons: *(i)* a long experimentation phase, which also allowed organizations with control over both end-points of a connection (Facebook and Google) to experiment with the protocol relatively risk-free and *(ii)* Internet centralization is the major factor in the fast deployment of the final version. The globally acting front-end provider Cloudflare is responsible for strong deployment across all our domain sets. For a number of countries, we also find regional centralization, *i.e.,* regionally operating, smaller providers activate TLS 1.3 for their customers.

The operational benefits of being able to control both endpoints of a connection are undeniable. This is particularly evident in the case of Facebook, who used custom versions to trial TLS 1.3, without negatively impacting access by third-party implementations via the web. We observe less experimentation for Google; but we note that this can be due to us excluding browsers from our Android data collection.

Our study highlights the importance of being able to draw on datasets from multiple sources. Compared to our previous work [9], our passive monitoring can no longer pick up on interesting artefacts such as TLS extensions or certificates, as these are encrypted. Active scans are a necessity. Also,

we would not have been able to identify operational setups like Facebook's without the data from both active scans and passive observation in the mobile ecosystem.

Further data could improve studies like ours even more. Aggregate server side statistics would provide insights about the used browsers, allowing us to address questions such as why some businesses and organizations are holding back on the deployment of TLS 1.3. This would also be a question worth following up with surveys or in-depth interviews. While Google publishes a Transparency Report, the granularity needed by researchers is far higher and requires adequate methods for privacy-preserving sharing. We would welcome a more official form of collaboration within the IETF or other bodies that allows access to such data. We would also posit that the fact that centralization is such a key driver for TLS 1.3 should serve as a call to the network community to contemplate the effects of centralization on the development of Internet technology at greater depth.

ACKNOWLEDGEMENTS

We are very grateful to our reviewers for their insightful comments and suggestions. We would like to thank Marwan Fayed in particular for reviewing in the open and engaging in highly constructive dialogue.

We thank VirusTotal for providing an academic licence for their API. We also thank ip2location for providing valuable assistance and an academic licence for their network categorization API.

This work was partially funded by the NSW Cybersecurity Network's Pilot Grant Program 2018 and a Major Equipment Grant by the Faculty of Engineering, University of Sydney. This work was supported by the US National Science Foundation under grants CNS-1528156 and CNS-1564329. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors or originators, and do not necessarily reflect the views of the National Science Foundation.

[1] [n.d.]. massdns. A high-performance DNS stub resolver in C. Fork of massdns by Quirin Scheitle. https://github.com/quirins/massdns.
[2] [n.d.]. OpenSSL changelog. https://www.openssl.org/news/changelog.html.
[3] [n.d.]. Zeek Network Security Monitor. https://www.zeek.org/.
[4] [n.d.]. zgrab. Go application layer scanner. Fork of zgrab. https://github.com/tls-evolution/zgrab.
[5] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, and Paul Zimmermann. 2015. Imperfect forward secrecy: how Diffie-Hellman fails in practice. In *Proc. ACM SIGSAC Conference on Computer and Communications Security (CCS)*.
[6] D. Akhawe, J. Amann, M. Vallentin, and R. Sommer. 2013. Here's my Cert, so trust me, maybe? Understanding TLS errors on the Web. In *Proc. of the International Web Conference (WWW)*.
[7] Alessandro Ghedini. 2018. You get TLS 1.3! You get TLS 1.3! Everyone gets TLS 1.3! https://blog.cloudflare.com/you-get-tls-1-3-you-get-tls-1-3-everyone-gets-tls-1-3/.
[8] N. J. AlFardan and K. G. Paterson. 2013. Lucky Thirteen: breaking the TLS and DTLS record protocols. In *Proc. IEEE Symposium on Security and Privacy (S&P)*.
[9] J. Amann, O. Gasser, Q. Scheitle, L. Brent, G. Carle, and R. Holz. 2017. Mission accomplished? HTTPS security after DigiNotar. In *Proc. ACM Internet Measurement Conference (IMC)*. London.
[10] Johanna Amann, Robin Sommer, Matthias Vallentin, and Seth Hall. 2013. No attack necessary: the surprising dynamics of SSL trust relationships. In *Proc. Annual Computer Security Applications Conference (ACSAC)*.
[11] J. Amann, M. Vallentin, S. Hall, and R. Sommer. 2012. *Extracting certificates from live traffic: a near real-time SSL notary service*. Technical Report TR-12-014. ICSI.
[12] Nimrod Aviram, Sebastian Schinzel, Juraj Somorovsky, Nadia Heninger, Maik Dankel, Jens Steube, Luke Valenta, David Adrian, J. Alex Halderman, Viktor Dukhovni, Emilia Käsper, Shaanan Cohney, Susanne Engels, Christof Paar, and Yuval Shavitt. 2016. DROWN: breaking TLS using SSLv2. In *Proc. USENIX Security Symposium*.
[13] Christian Badertscher, Christian Matt, Ueli Maurer, Phillip Rogaway, and Björn Tackmann. 2015. Augmented secure channels and the goal of the TLS 1.3 record layer. In *International Conference on Provable Security*. Springer, 85–104.
[14] Mihir Bellare and Björn Tackmann. 2016. The multi-user security of authenticated encryption: AES-GCM in TLS 1.3. In *Annual International Cryptology Conference (CRYPTO)*. Springer.
[15] David Benjamin. 01 Jun 2016. [TLS] Downgrade protection, fallbacks, and server time. https://mailarchive.ietf.org/arch/msg/tls/la87rmkU4Ay0hrhyiddmbjOQRsY/.
[16] David Benjamin. 11 Jan 2018. TLS ecosystem woes. Talk at RWC 2018. https://youtu.be/_mE_JmwFi1Y?t=1167.
[17] David Benjamin. 18 Dec 2017. Additional TLS 1.3 results from Chrome. Post to IETF mailing list. https://mailarchive.ietf.org/arch/msg/tls/i9blmvG2BEPf1s1OJkenHknRw9c/.
[18] D. Benjamin. 2019. Applying GREASE to TLS extensibility. https://tools.ietf.org/html/draft-ietf-tls-grease-02
[19] D. Benjamin. 2020. Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS extensibility. https://tools.ietf.org/html/rfc8701
[20] Benjamin Beurdouche, Antoine Delignat-Lavaud, Nadim Kobeissi, Alfredo Pironti, and Karthikeyan Bhargavan. 2015. FLEXTLS: a tool for testing TLS implementations. In *USENIX Workshop on Offensive Technologies (WOOT)*.
[21] Karthikeyan Bhargavan, Bruno Blanchet, and Nadim Kobeissi. 2017. Verified models and reference implementations for the TLS 1.3 standard candidate. In *Proc. IEEE Symposium on Security and Privacy (S&P)*.
[22] Google blog. 2019. Android Q features and APIs. https://android-developers.googleblog.com/2019/03/introducing-android-q-beta.html.
[23] L. Chuat, P. Szalachowski, A. Perrig, B. Laurie, and E. Messeri. 2015. Efficient gossip protocols for verifying the consistency of certificate logs. In *IEEE Conference on Communications and Network Security (CNS)*.
[24] J. Clark and P. van Oorschot. 2013. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In *Proc. IEEE Symposium on Security and Privacy (S&P)*.
[25] Blue Coat. 2017. ProxySG, ASG and WSS will interrupt SSL connections when clients using TLS 1.3 access sites also using TLS 1.3. https://web.archive.org/web/20170912061432/http://bluecoat.force.com/knowledgebase/articles/Technical_Alert/000032878.
[26] Cas Cremers, Marko Horvat, Jonathan Hoyland, Sam Scott, and Thyla van der Merwe. 2017. A comprehensive symbolic analysis of TLS 1.3. In *Proc. ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM.
[27] Antoine Delignat-Lavaud, Cédric Fournet, Markulf Kohlweiss, Jonathan Protzenko, Aseem Rastogi, Nikhil Swamy, Santiago Zanella-Béguelin, Karthikeyan Bhargavan, Jianyang Pan, and Jean Karim Zinzindohoue. 2017. Implementing and proving the TLS 1.3 record layer. In *Proc. IEEE Symposium on Security and Privacy (S&P)*. IEEE, 463–482.
[28] David Dittrich and Erin Kenneally. 2012. *The Menlo Report: Ethical principles guiding information and communication technology research*. Technical Report. US Department of Homeland Security.
[29] Benjamin Dowling, Marc Fischlin, Felix Günther, and Douglas Stebila. 2015. A cryptographic analysis of the TLS 1.3 handshake protocol candidates. In *Proc. ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM.

[30] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. 2013. ZMap: Fast Internet-wide scanning and its security applications. In *Proc. USENIX Security Symposium*.
[31] Facebook. 2018. Deploying TLS 1.3 at scale with Fizz, a performant open source TLS library. https://code.fb.com/security/fizz/.
[32] Josef Gustafsson, Gustaf Overier, Martin Arlitt, and Niklas Carlsson. 2017. A first look at the CT landscape: Certificate Transparency logs in practice. In *Proc. Passive and Active Measurement (PAM)*.
[33] T. Halvorson, M. F. Der, I. Foster, S. Savage, L. K. Saul, and G. M Voelker. 2015. From .academy to .zone: an analysis of the new TLD land rush. In *Proc. ACM Internet Measurement Conference (IMC)*. Tokyo.
[34] Ralph Holz, Johanna Amann, Olivier Mehani, Matthias Wachs, and Mohamed Ali Kaafar. 2016. TLS in the wild: An Internet-wide analysis of TLS-based protocols for electronic communication. In *Proc. Network and Distributed System Security Symposium (NDSS)*.
[35] Ralph Holz, Lothar Braun, Nils Kammenhuber, and Georg Carle. 2011. The SSL Landscape: a thorough analysis of the X.509 PKI using active and passive measurements. In *Proc. ACM Internet Measurement Conference (IMC)*.
[36] Tibor Jager, Jörg Schwenk, and Juraj Somorovsky. 2015. On the security of TLS 1.3 and QUIC against weaknesses in PKCS#1 v1.5 encryption. In *Proc. ACM SIGSAC Conference on Computer and Communications Security (CCS)*.
[37] Nadim Kobeissi. 2018. *Formal Verification for Real-World Cryptographic Protocols and Implementations*. Ph.D. Dissertation. INRIA Paris; École Normale Supérieure de Paris—ENS Paris.
[38] Platon Kotzias, Abbas Razaghpanah, Johanna Amann, Kenneth G Paterson, Narseo Vallina-Rodriguez, and Juan Caballero. 2018. Coming of age: a longitudinal study of TLS deployment. In *Proc. ACM Internet Measurement Conference (IMC)*.
[39] Hugo Krawczyk and Hoeteck Wee. 2016. The OPTLS protocol and TLS 1.3. In *Proc. IEEE European Symposium on Security and Privacy (EuroS&P)*.
[40] Bodo Möller. 2014. This POODLE bites: exploiting the SSL 3.0 fallback. Google blog. https://security.googleblog.com/2014/10/this-poodle-bites-exploiting-ssl-30.html.
[41] Nick Sullivan. 2016. Introducing TLS 1.3. https://blog.cloudflare.com/introducing-tls-1-3/.
[42] Nick Sullivan. 2017. Introducing Zero Round Trip Time Resumption (0-RTT). https://blog.cloudflare.com/introducing-0-rtt/.
[43] Nick Sullivan. 2017. Why TLS 1.3 isn't in browsers yet. https://blog.cloudflare.com/why-tls-1-3-isnt-in-browsers-yet/.
[44] Craig Partridge and Mark Allman. 2016. Addressing ethical considerations in network measurement papers. *Commun. ACM* 59, 10 (Oct. 2016), 58–64.
[45] Abbas Razaghpanah, Arian Akhavan Niaki, Narseo Vallina-Rodriguez, Srikanth Sundaresan, Johanna Amann, and Phillipa Gill. 2017. Studying TLS usage in Android apps. In *Proc. ACM Int. Conference on emerging Networking EXperiments and Technologies (CoNEXT)*.
[46] Abbas Razaghpanah, Narseo Vallina-Rodriguez, Srikanth Sundaresan, Christian Kreibich, Phillipa Gill, Mark Allman, and Vern Paxson. 2015. Haystack: A multi-purpose mobile vantage point in user space. https://arxiv.org/abs/1510.01419.
[47] Eric Rescorla. 2016. The Transport Layer Security (TLS) Protocol Version 1.3 - Draft 16. https://tools.ietf.org/html/draft-ietf-tls-tls13-16
[48] Eric Rescorla. 2017. The Transport Layer Security (TLS) Protocol Version 1.3 - Draft 22. https://tools.ietf.org/html/draft-ietf-tls-tls13-22
[49] Eric Rescorla. 2018. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446 (Historic). https://tools.ietf.org/html/rfc8446 RFC 8446.
[50] Eric Rescorla. 2018. The Transport Layer Security (TLS) Protocol Version 1.3 - Draft 28. https://tools.ietf.org/html/draft-ietf-tls-tls13-28
[51] Mark D. Ryan. 2014. Enhanced Certificate Transparency and end-to-end encrypted mail. In *Proc. Network and Distributed System Security Symposium (NDSS)*.
[52] Quirin Scheitle, Taejoong Chung, Johanna Amann, Oliver Gasser, Lexi Brent, Georg Carle, Ralph Holz, Jens Hiller, Johannes Naab, Roland van Rijswijk-Deij, et al. 2018. Measuring adoption of security additions to the HTTPS ecosystem. In *Proc. Applied Networking Research Workshop*.
[53] Quirin Scheitle, Oliver Gasser, Theodor Nolte, Johanna Amann, Lexi Brent, Georg Carle, Ralph Holz, Thomas C Schmidt, and Matthias Wählisch. 2018. The rise of Certificate Transparency and its implications on the Internet ecosystem. In *Proc. ACM Internet Measurement Conference (IMC)*.
[54] Quirin Scheitle, Oliver Hohlfeld, Julien Gamba, Jonas Jelten, Torsten Zimmermann, Stephen D. Strowes, and Narseo Vallina-Rodriguez. 2018. A long way to the top: significance, structure, and stability of Internet top lists. In *Proc. ACM Internet Measurement Conference (IMC)*.
[55] B. VanderSloot, J. Amann, M. Bernhard, Z. Durumeric, M. Bailey, and J.A. Halderman. 2016. Towards a complete view of the certificate ecosystem. In *Proc. ACM Internet Measurement Conference (IMC)*.
[56] Scott Yilek, Eric Rescorla, Hovav Shacham, Brandon Enright, and Stefan Savage. 2009. When private keys are public: results from the 2008 Debian OpenSSL vulnerability. In *Proc. ACM Internet Measurement Conference (IMC)*.
[57] L. Zhang, D. Choffnes, D. Levin, T. Dumitras, A. Mislove, A. Schulman, and C. Wilson. 2014. Analysis of SSL certificate reissues and revocations in the wake of Heartbleed. In *Proc. ACM Internet Measurement Conference (IMC)*.

A ETHICAL CONSIDERATIONS

Our study involves the passive collection of network traffic from real users and active network scans. We follow the principles of informed consent [28] and best practices [44]: we avoid the collection of any personal or sensitive data, such as client IP addresses or traffic payloads, and we try to avoid causing any harm to online servers during our active scans. Below we discuss details specific to each tool.

A.1 Passive data collection

The passive data collection effort performed by the ICSI SSL Notary was cleared by the respective responsible parties at each contributing institution before they began contributing. Note that the ICSI SSL Notary specifically excludes or anonymizes sensitive information, such as client IP addresses. In more detail, client IP addresses are combined with the server IP address and SNI as well as a site-specific secret unknown to ICSI. The resulting string is hashed. This allows the dection of when the same client connected to the same IP address (*e.g.,* to evaluate the effectiveness of session resumption), without enabling the tracking of a client while it accesses different servers. It also means that ICSI data does not contain any information of how many users are active at a specific site. While the Notary records server-sent certificates, it does not record client certificates if they are present in the handshake. The Notary only records handshake information that is sent in the clear.

A.2 Active scans

We took precautions to minimize the impact of our scans, following established practices as, for instance, described in [30]. In particular, we maintain a blacklist to avoid scanning systems that have in the past indicated to us that they do not wish to be scanned. Our abuse email address is published in the WHOIS and all abuse emails are forwarded to us by our IT department. We received one abuse email sent by a blocklist provider; our scanner was whitelisted when we explained our work. Our scanning activity was also reviewed by the Human

Ethics board of our hosting institution; it was found that we do not collect personally identifiable information and hence need not undergo a Human Ethics approval process. We assess the impact of our scans in terms of potential harm to other systems and human beings, as proposed by the Menlo report [28]. We use a relatively low scanning rate to minimize any impact and respond immediately to complaints.

A.3 Lumen Privacy Monitor

Lumen's view of real-world mobile data collected from end-user devices raises ethical issues. We address these in two ways:

Informed Consent. Lumen follows the principles of informed consent as indicated by the Menlo Report [28] and avoids the collection of any personal or sensitive data. Users must explicitly grant permission to Lumen to inspect the traffic and the app requires users to opt-in a second time to install a CA certificate to inspect encrypted traffic. Furthermore, the user can disable traffic interception and uninstall the app at any time. The privacy policy of the app is available in Google Play as well as on the project's website:

https://haystack.mobi/privacy.html

Data Collection Strategy. Lumen runs on the user's device. It allows Lumen to confine the bulk of the data processing to the device itself. Lumen only collects and uploads anonymized summary statistics to the project servers. Mobile app traffic flows are mapped to the app generating them and not to a user identity. For example, we collect flow metadata like TLS *ClientHello* and *ServerHello* records, HTTP User Agent Field, byte counts, the destination IP address and TCP port number, the package name and version of the app making the connection, and the Android version on the device.

The data is uploaded following reasonable security mechanisms (*i.e.,* use of encryption). To further protect user privacy, Lumen also ignores flows generated by applications which may potentially deanonymize a user. Examples of such applications are mobile browsers such as the *Android default browser* or *Google Chrome*. The type of traffic generated by these apps is highly dependent on user actions, which not only makes deanonymizing users easier but also beats our purpose of understanding the way that mobile apps work due to developer decisions. The team behind Lumen follows ethical protocols, which were developed in consultation with their Institutional Review Board (IRB)—Lumen is considered a non-human subject research effort due to the anonymization process—before starting any data collection. The portion of Lumen's data that was used in this study and the scripts used to analyze the data are available on the project's website:

https://haystack.mobi/datasets

B REPRODUCTION OF OUR RESEARCH

We aim to enable reproducibility of our results. In the following, we discuss how we achieve *repeatability* (the research can be repeated by the same reviewers with the same experimental setup), *replicability* (the research can be performed by a different team with the same experimental setup), and *reproducibility* (a different team is able to reproduce our results, within the limitations explained next, using a suitable experimental setup of their own devising).

B.1 Repeatability

As the Internet is rapidly changing, especially with respect to the support of an evolving new standard, repeated Internet measurements will not yield the same results. Thus, there is a natural limit for repeatability of our research. Nevertheless, to minimize the risk of errors and one-time effects, we repeated our measurements over several months and store raw or slightly prefiltered data. We apply the same scripts to extract information.

B.2 Replicability

To make our research replicable, we release the code of our scanners, our analysis scripts, and—where possible—all measurement results. We release all data from active scans when no legal limitations exist. Some active measurements, however, relied on information provided to us under NDAs, *e.g.,* specific top-level domain lists, which prevents us from publishing all raw data. In some cases, we provide aggregated data that enables checking of our results while not violating NDAs. If a different team should enter into the same NDAs, they will be able to get access to the data. Our code and analysis scripts will then allow them to run the full analysis.

Specifically, we release our data of active RFC scans in both raw (PCAP traces) and processed format (CSV). For our scans of draft versions, we generally provide processed data (JSON). We cannot provide the raw data (JSON output of *zgrab*) except for the results for the Alexa Top1M. We also provide data from the Lumen collection (see Appendix A.3). We cannot release data from passive monitoring at the Notary for ethical and legal reasons.

We provide detailed information, our scanner and analysis scripts, as well as measurement data under:

https://tls13.globalsecuritylabs.org

B.3 Reproducibility

To enable reproducibility of our research with a different experimental setup, we provide information that explains how to set up a scanning or monitoring environment. Specifically, we provide guidance how to obtain the relevant information for scanning or monitoring, how to follow ethical guidelines, and how to analyze the results. As with the other data, this information is provided in the repository given above.