Mission Accomplished? HTTPS Security after DigiNotar

Johanna Amann[*]
ICSI / Corelight / LBNL
johanna@icir.org

Oliver Gasser[*]
Technical University of Munich
gasser@net.in.tum.de

Quirin Scheitle[*]
Technical University of Munich
scheitle@net.in.tum.de

Lexi Brent
The University of Sydney
lexi.brent@sydney.edu.au

Georg Carle
Technical University of Munich
carle@net.in.tum.de

Ralph Holz
The University of Sydney
ralph.holz@sydney.edu.au

ABSTRACT

Driven by CA compromises and the risk of man-in-the-middle attacks, new security features have been added to TLS, HTTPS, and the web PKI over the past five years. These include Certificate Transparency (CT), for making the CA system auditable; HSTS and HPKP headers, to harden the HTTPS posture of a domain; the DNS-based extensions CAA and TLSA, for control over certificate issuance and pinning; and SCSV, for protocol downgrade protection.

This paper presents the first large scale investigation of these improvements to the HTTPS ecosystem, explicitly accounting for their combined usage. In addition to collecting passive measurements at the Internet uplinks of large University networks on three continents, we perform the largest domain-based active Internet scan to date, covering 193M domains. Furthermore, we track the long-term deployment history of new TLS security features by leveraging passive observations dating back to 2012.

We find that while deployment of new security features has picked up in general, only SCSV (49M domains) and CT (7M domains) have gained enough momentum to improve the overall security of HTTPS. Features with higher complexity, such as HPKP, are deployed scarcely and often incorrectly. Our empirical findings are placed in the context of risk, deployment effort, and benefit of these new technologies, and actionable steps for improvement are proposed. We cross-correlate use of features and find some techniques with significant correlation in deployment. We support reproducible research and publicly release data and code.

CCS CONCEPTS

• **Security and privacy** → **Network security**;

[*]Joint first authorship.

ACM Reference Format:
Johanna Amann, Oliver Gasser, Quirin Scheitle, Lexi Brent, Georg Carle, and Ralph Holz. 2017. Mission Accomplished? HTTPS Security after DigiNotar. In *Proceedings of IMC '17, London, United Kingdom, November 1–3, 2017,* 16 pages.
https://doi.org/10.1145/3131365.3131401

1 INTRODUCTION

The compromise of the DigiNotar CA in 2011 [57] was a decisive event in the history of WWW security: for the first time, a CA was removed from browser root stores, because of poor infrastructure control and the subsequent issuance of forged certificates [51]. In the same year, several studies documented the poor state of the TLS/X.509 ecosystem in general [8, 24, 39].

Since then, a number of improvements and additions have been proposed to strengthen the X.509 PKI. They include Certificate Transparency (CT) [44], which establishes a set of publicly verifiable append-only certificate logs; HTTP Strict Transport Security (HSTS) [36], which instructs browsers to only connect through HTTPS; HPKP, which allows certificate pinning through HTTP headers [25]; SCSV, which protects against protocol downgrading attacks [50]; DANE-TLSA, which enables HTTPS certificate pinning using the DNS [37]; and finally CAA [33], which allows control of certificate issuance through the DNS.

In this paper, we investigate the prevalence of these technologies, audit the correctness of their deployment, and examine the *combined* role they play in post-DigiNotar web security. Our contributions are as follows: We combine active and passive measurements to investigate the improvements to TLS and the web PKI. Our measurements include hitherto the largest active scans of the TLS/X.509 ecosystem. Rather than performing an active scan of the IP address space, our scans target a list of 193M domain names. This provides a more complete view, as many HTTPS servers implement the *Server Name Indication (SNI)* extension to serve different certificates and use different TLS configurations per domain [71]. This domain-based approach also reduces bias caused by the common presence of embedded devices that happen to run web servers [16]. Our active scans originate from two vantage points on opposite ends of the globe, using both IPv4 and IPv6. We also perform passive measurements in North America, Europe, and Australia. To the best of our knowledge, this is the first time that such a geographically diverse passive TLS observation has been conducted. Our data analysis uses a novel process in which active and passive data share the same analysis pipeline.

We investigate each of the above technologies in depth, particularly focusing on Certificate Transparency and new TLS and HTTPS extensions. Importantly, we also investigate how these technologies are used *in combination*, and which protection level is thereby achieved. We paint an accurate picture to which degree these technologies are correctly deployed and which mistakes are commonly made. We contextualize our empirical findings and explore the correlation between complexity, benefit, and risk of each technology. Additionally, we examine the proliferation of different TLS versions by drawing on a massive data set of global connection-level TLS information collected since 2012.

We strive to support open science and release our active scan dataset to the community. Along with parsed results, we make packet-level data captures available, allowing for precise reproduction and new uses. We also feed software changes back to the community, and publish newly created tools under a permissive open-source license. Data and code can be found at https://mediatum.ub.tum.de/1377982.

The remainder of this paper is organized as follows. Section 2 covers the technical background. Section 3 details the related work. Section 4 describes our methodology. Sections 5, 6, 7, and 8 present our results for CT, HSTS and HPKP, SCSV, and the DNS-based extensions CAA and DANE-TLSA, respectively. Section 9 shows the evolution of TLS version use over the last five years. We discuss our findings and relate them to risk, cost, and benefit of the new technologies in Section 10, and summarize them in Section 11.

2 BACKGROUND

This section describes the TLS, HTTP, and DNS based HTTPS security extensions we investigate. For a general web PKI introduction, we refer the reader to [18, 39].

CT: Certificate Transparency (CT) [44] aims to make unnoticed attacks on the PKI near-impossible through public disclosure of certificate issuance. Users or CAs submit certificate chains for inclusion in one or more semi-trusted public logs, run by independent parties. Each log stores entries in an append-only Merkle Hash Tree. Observers can detect tampering or holes in the dataset by requesting consistency proofs from the log. A goal of CT is for browsers to display lower security indicators if certificates are not logged; currently Google Chrome is the only browser performing this step.

Upon submission of a certificate chain, a log server returns a signed promise of inclusion called the Signed Certificate Timestamp (SCT). The SCT can be verified using the log's public key. TLS servers deliver SCTs to the browser, either embedded in the certificate as an X.509 extension, via a TLS extension, or in an Online Certificate Status Protocol (OCSP) response delivered as a TLS extension (OCSP stapling).

To embed SCTs in a certificate, the CA submits a signed precertificate to the CT logs. This precertificate is a promise that a CA is going to issue this exact certificate after receiving the SCT. The precertificate contains a *poison* extension that prevents browsers from validating it; it cannot be used in place of a real certificate. The log server signs the precertificate and returns SCTs for it. These are embedded into an X.509 extension of the final certificate. Browsers verify the embedded SCTs by reconstructing the precertificate.

At the time of writing, Google Chrome is the only popular browser that verifies SCTs. It supports all transmission methods, and requires valid SCTs for Extended Validation (EV) certificates, removing the EV trust indicator otherwise.

HTTP-based extensions: HTTP Strict Transport Security (HSTS) [36] and HTTP Public Key Pinning (HPKP) [25] are HTTP extensions that aim to increase the security of the HTTPS ecosystem by setting HTTP header values. HSTS instructs the client to only access a domain via HTTPS. HPKP enables the server to pin specific public keys to a domain to mitigate man-in-the-middle attacks. Browsers must abort a connection if none of the pins match the certificate chain used by the domain. Both HSTS and HPKP directives are shipped with web browsers in so-called preloading lists.

SCSV Downgrade Prevention: RFC 7507 [50] defines a *Signaling Cipher Suite Value* (SCSV) that is used to prevent downgrade attacks in which an attacker prevents connections with strong TLS versions in order to exploit weaknesses in older TLS versions. Clients commonly fall back to older TLS versions if a connection attempt with a newer TLS version is unsuccessful. In this fallback case, the client appends the SCSV pseudo-cipher value to its list of supported ciphers. When receiving this SCSV, the server must abort the connection if it supports a higher protocol version. One motivation for SCSV was the infamous POODLE attack [49].

DNS-based Extensions: Both Certification Authority Authorization (CAA) [33] and TLS Authentication (TLSA) [37] are DNS record types introduced to aid certificate issuance and verification, respectively. CAA indicates which CAs may issue certificates for a domain. It also supports reporting in cases where a CA is requested to issue a certificate for a domain, but may not do so because of the CAA record. CAA was accepted by the CA/Browser forum as a mandatory step during certificate issuance [12] and became effective on September 8, 2017.

In contrast to all other methods, the CAA record is only required to be correct at the time a CA issues a certificate. Browsers must not match it against current certificates. TLSA, which mandates DNSSEC, allows domain owners to specify which certificate or public key is meant to be deployed for a specific domain and port.

3 RELATED WORK

Our research stands in the line of a large body of work on the TLS and HTTPS ecosystem. Past studies have analyzed different segments of the ecosystem and highlighted its many shortcomings, for example, focusing on the PKI [3, 7, 24, 39], communication protocols [23, 38], certificate revocation [79–81], cryptographic properties and weaknesses [2, 35, 40], and implementation problems [10]. For a thorough explanation and review of the web PKI and its weaknesses, we refer to [18, 39].

Most closely related to our work, VanderSloot *et al.* [71] examine the HTTPS ecosystem from several perspectives, including active scans, passive monitoring and Certificate Transparency logs. While they examine CT, they only focus on some of the differences observed in CT and other scan methods. They do not examine the properties of CT extensively, as we do in this work.

Several papers examine ways to enhance or optimize Certificate Transparency. Chuat *et al.* [15] explore ways to exchange CT information via gossiping to defend against logs lying to small groups of users. Ryan [62] extends CT to certificate revocation and addresses end-to-end encrypted email.

Gustafsson *et al.* [32] present a study that analyzes the content of CT logs. The authors focus on data directly obtained from the logs and show differences between smaller CA-owned logs and Google's large logs filled from Internet scans. They track how many of the domain names obtained from monitoring their campus network match certificates in logs. In contrast, this study is more extensive: it includes data from active and passive observations and provides an in-detail analysis of both SCTs and certificates.

Clark and van Oorshot [18] theoretically studied the effects of HTTP extension headers in 2012. Kranch and Bonneau [42] study the deployment of HSTS and HPKP based on both the preload and the Alexa Top 1M lists. De los Santos *et al.* [21] analyze the implementation of HSTS and HPKP for several dozen domains using Shodan. Given the novelty of both standards, we find the uptake of HSTS and HPKP to have significantly changed since these early studies.

Although there is a large body of DNSSEC measurement work [46, 53], interest in TLSA and CAA has been limited. An early study in 2014 shows very low deployment of TLSA records, reporting less than 1000 records in the *.com* and *.net* zones [82]. Our study shows that no explosive growth has taken place since then. Szalachowski and Perrig [68] count deployment of CAA and TLSA among the Alexa Top 100k domains which we compare against in Section 8.

4 METHODOLOGY

Our work combines active scans and passive measurements from several sites. We collect the following data. For **CT**, we extract Signed Certificate Timestamps (SCTs) from X.509 certificates as well as TLS and OCSP extensions. A modified version of Google's log monitor software [29, 31] is used to retrieve certificates from logs accepted by the Google Chrome browser (as of May 2017) [30].

For **HSTS/HPKP**, we parse and analyze the HTTP responses our scanner collects. For **SCSV**, we lower the TLS version and set the Signaling Cipher Suite Value for Downgrade Protection. This should cause clients to reject the connection (see Section 2). We collect **CAA** and **TLSA** resource records from DNS. Details are provided in each section.

We devise a novel way to unify the processing of data from active scans and passive network monitoring to analyze CT properties. We dump the raw network traffic of the active scan into a *pcap* trace. This trace is then fed into our passive measurement pipeline. By using the same analysis code paths for active and passive data, we achieve full comparability. This also enables us to share the *raw* data from active scans. In contrast to earlier work, which shares *processed* data from active scans, this packet-level information allows better reproducibility and exposes information about exact packet timing and timeout behavior.

4.1 Active Scans

We conduct active scans from the University of Sydney (IPv4), and the Technical University of Munich (IPv4 & IPv6). As discussed in Section 1, our scan is based on domain names as opposed to IP addresses. This captures SNI-based servers (cf. [16, 71]) and avoids accidentally connected devices.

We note that TLS scans based on domain names have been carried out before, often using the Alexa Top 1 million list of popular domains. In 2016, VanderSloot *et al.* [71] used *.com*, *.net*, and *.org* domains to scan 153M domains. We extend this approach by adding domains from *.biz*, *.info*, *.mobi*, *.sk*, and *.xxx* from PremiumDrops [56]; *.de* and *.au* from ViewDNS [74]; from the Alexa [4] and Umbrella [17] Top 1M, all Alexa Country Top 50 [5], plus domains from 748 zones from ICANN's Centralized Zone Data Service [41]. This yields a total of 193M domain names, about 58% of the 330.6M registered domains in March 2017 [72].

We resolve domains from both Munich (TUM) and Sydney (USyd) using a modified version of *massdns* [11] and an unmodified version of *unbound* [43]. From Munich, we find 154M IPv4-enabled and 9.7M IPv6-enabled domains, with a 9.5M intersection. From Sydney, we considered only A records, as the university network does not support IPv6. 650k (0.4%) less domains could be resolved. This is within expectations: Rijswijk-Deij *et al.* [70] show that daily deviations of around 0.6% are expected for large-scale DNS scans. IP addresses learned from our DNS scans are port-scanned using a custom [83] IPv6 capable version of ZMap [24]. The IPv6 response rate is in line with previous work [27]. We perform TLS handshakes using Goscanner [26], a custom highly-parallelized scanning tool. Goscanner connects to each IP address, sending the domain name in the SNI extension, one name per connection.

If we can establish a TLS connection, we send an HTTP *HEAD* request to obtain HSTS and HPKP headers. In about 50% of cases, we receive an HTTP 200 ('OK') response code. In the remaining cases, we receive mainly redirect codes, error codes, or no HTTP response at all. For cases where the TLS handshake succeeds, we immediately connect a second time, offering a lower protocol version while sending the SCSV pseudo-cipher. This tests for downgrade protection on server side: servers should abort such connections.

Table 1 provides an overview of scan results along our scanning chain, across locations and protocols.

# of	TUM IPv4	USyd IPv4	TUM IPv6	Rel. Work
Input Domains	192.9M	192.9M	192.9M	≈153M [70, 71]
Domains $\geq$ 1 RR[1]	153.5M	152.9M	9.7M	149M [70]
IP addresses	8.8M	8.9M	6.2M	
tcp443 SYN-ACKs	4.0M	3.2M	316k	249k [27]
<domain,IP> pairs	80.4M	79.2M	11.0M	
Successful TLS SNI[2]	55.7M	58.0M	5.1M	42M [71]
HTTP response 200 SNIs	28.4M	28.1M	1.9M	

1: Domains that server 1 or more Resource Records of A or AAAA type.
2: <Domain,IP> tuples with successful TLS SNI connections.

Table 1: Overview of DNS Resolutions and Active Scans, conducted from April 11 through April 16, 2017.

Location	Time	TLS Conns.	Certs.	Valid
Berkeley	2.4.–2.5.	2.6G	1.5M	366.2k
port 443		2.5G	729.1k	364.2k
Munich	12.5.–16.5.	286.7M	178.7k	167.1k
Sydney	12.5.–16.5.	196.2M	115.8k	113k

Table 2: Overview of passive monitoring data. UC Berkeley is not filtered for port 443. Many autogenerated certificates, such as for WebRTC, drive certificate count for non-443.

4.2 Passive Monitoring

Our passive measurements use two data sources: *(i)* to measure the current use of Certificate Transparency in the Internet, we passively monitor the Internet uplinks of the University of California at Berkeley (UCB) for several weeks. To validate our results, we monitor the Internet uplinks at the University of Sydney (Australia) and the Technical University of Munich (Germany), about two weeks after the first monitoring run. *(ii)* For our TLS version evolution study we use data from the ICSI SSL Notary [8], which contains connection-level information from select research institutions and Universities starting in 2012.

We use the Bro Network Security Monitor [55] for our passive data analysis. For this work, we extend Bro to support TLS version 1.3, improve the OCSP support, and implement support for parsing and live-validation of CT information from certificates, TLS extensions and OCSP replies. The code is merged into Bro and will be part of Bro 2.6. In all environments, we only analyze outgoing connections to prevent bias from our internal server population. Table 2 shows details on the number of TLS handshakes and unique hosts seen at our three passive monitoring vantage points.

The University of Sydney has a 10GE Internet uplink. Port 443 traffic is mirrored to a 64-core Linux machine using PF_Ring for traffic distribution (4 AMD Opteron 6276 2.6GHz CPUs, 64GB RAM, Ubuntu 14.04.5). Only traffic from the Internet to the University network is mirrored to the analysis machine. Due to the fact that all CT information is contained in the server handshake, this still allows us to use data from Sydney for nearly all of our measurements. Readers familiar with Bro might be aware that Bro usually does not work well with one-sided traffic. This is not a limitation of the Bro core but of many Bro protocol analyzers. The TLS analyzer, however, parses one-sided TLS traffic correctly. We verified this by checking the TCP reassembler source code and creating several test-cases, available at [6].

The Technical University of Munich monitors the Munich Scientific Network's 2x10GE Internet uplink. Port 443 traffic is mirrored to a 24-core machine, again using PF_Ring (2 Intel Xeon E5-2630v2 2.6GHz, 256GB RAM, SLES12). Owing to the fact that the entire traffic is sent to the analysis machine over a single 10GE connection, packet-loss occurs during peak-times. This should not impact our analysis as packet loss should be uniformly distributed; furthermore we only examine the first few packets of every TLS connection. Our data analysis does indeed not show significant deviations for Munich.

UCB has a 10GE Internet uplink. The traffic is split up to 28 independent machines, each with 4 Intel Xeon 5430 CPUs at 2.66GHz and 12GB of RAM (FreeBSD 11.0). Traffic is distributed to 2 processes per machine using netmap [60] and lb [61]. Traffic is not filtered for port 443.

4.3 Ethical Considerations

For active scans, we minimize interference by following best scanning practices, such as those outlined in [24], by maintaining a blacklist and using dedicated servers with informing rDNS names, websites, and abuse contacts. We assess whether data collection can harm individuals or reveal private information as proposed by [22, 54]. Our passive data collection was cleared by the responsible parties at each contributing institution; this includes all institutions contributing data to the ICSI SSL Notary. Note that our passive data collection specifically excludes or anonymizes sensitive information, such as client IP addresses.

5 CERTIFICATE TRANSPARENCY

To evaluate the use of Certificate Transparency, we extract and validate SCTs from both active scans and passive observations using our extended version of Bro (see Section 4). As detailed in Section 2, SCTs can be retrieved via three different ways. SCTs received via the TLS extension or OCSP stapling are directly validated using the server certificate and the log public key. In the case of SCTs embedded in an X.509 certificate, the reconstructed precertificate is used in place of the end-entity certificate, as specified in [44] (see Section 2). In this case, the verification step also requires information from the CA certificate issuing the end host certificate. Specifically, the issuer key hash of the parent certificate is included in the signed data structure [44]; this information can only be obtained from the CA certificate.

We determine the issuer CA certificate needed for SCT validation in a multi-step process. First, validation of the presented chain is attempted against Mozilla's root store using a process similar to that of Firefox, caching certificates from previous connections [3]. This enables us to validate server certificates and SCTs even if root or intermediate CA certificates are missing from the connection. If this step fails,

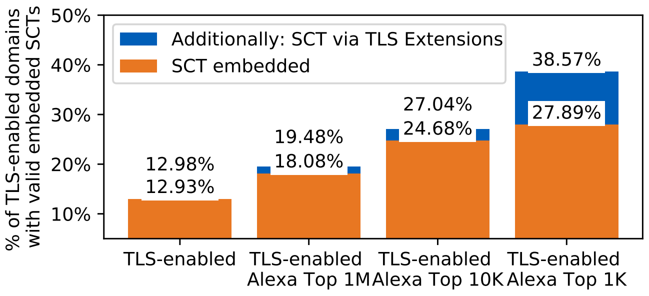

Figure 1: Embedded SCTs on domains. Blue bar represents domains using SCT via TLS extensions that did not already serve SCT via X.509.

SCT validation is attempted using each certificate present in the connection.

5.1 Measurement Results

Table 3 gives an overview of the CT data obtained in our active scans. We count a domain as supporting CT if at least one connection to one of its IP addresses transported SCTs. The IPv4 scans from Munich and Sydney show very similar numbers of connections with SCTs present—on the order of 7.6M connections, *i.e.,* between 12.7% and 13.3% of all TLS connections in which we receive a certificate from the server. This translates to about 6.8M domains that support CT. The number of domains supporting CT via IPv6 is, in line with IPv6 deployment, much lower (357k).

Figure 1 shows greater use of CT among popular domains, which are also more likely to transmit SCTs via TLS extensions. We speculate this might be an effort to optimize mobile experiences: SCTs are only sent by TLS extensions if requested by the client. Not including them in the certificate saves several 100 bytes at the beginning of mobile HTTPS transactions, which typically do not support CT.

	All	SYDv4	MUCv4	MUCv6
Domains w/ SCT	7.0M	6.8M	6.8M	357k
via X.509	7.0M	6.7M	6.8M	344k
via TLS	27.8k	27.6k	27.2k	12.9k
via OCSP	191	180	188	3
Operator diversity[1]	6.9M	6.7M	6.7M	349k
Certificates	11.69M	10.62M	9.66M	594.98k
with SCT	868.5k	800.5k	835.3k	194.2k
via X.509	867.6k	799.9k	834.5k	193.9k
via TLS	885	631	759	346
via OCSP	49	43	47	3
Valid EV Certs	66k	64.1k	62.9k	2.1k
with SCT	65.6k	63.6k	62.5k	2.1k
without SCT	459	451	436	3

1: Cert. logged in at least one Google & one non-Google operated log.

Table 3: CT data from active scans.

In fact, the results from our different vantage points are all very similar. Combined, we receive a total of 11.7M certificates in our active scans. We find that SCTs are almost exclusively embedded in X.509 certificates: less than a thousand are received in the TLS extension, and not even 50 in OCSP staples. Although less than 7.5% of certificates contained SCTs, we consider this an impressive fraction, showing significant CT deployment within a short time frame. Furthermore, almost all domains with CT support present one SCT from a Google-operated log and one from a non-Google log—this is the minimum requirement for Chrome to recognize a certificate's EV status [28] (although the number of EV certificates is actually very small). Interestingly, the almost exclusive preference for embedded certificates stands in contrast to initial plans of many CAs, which indicated a preference for delivering SCTs via OCSP [13].

49 certificates where the SCT was sent in an OCSP staple were issued by SwissSign (30), DigiCert (17), and Comodo (2). 7 of the DigiCert certificates were issued for Dropbox domains, and the Comodo certificates were issued for *sslanalyzer.comodoca.com* and *medicalchannel.com.au*. This demonstrates that CAs do not commonly enable SCT delivery via OCSP, possibly even only on customer request.

Extended Validation certificates contain SCTs in more than 99% of cases. This is probably explained by the Chrome EV policy [28] which requires SCTs to be presented for EV certificates in order for the 'green bar' to be displayed. Of the certificates missing an SCT, 164 were issued by Verizon Enterprise Solutions, 86 by Symantec, and 77 by Certplus.

The above findings are corroborated by our passive data, shown in Table 4. Ratios between data points are relatively similar, independent of the vantage point. In the following, we will refer primarily to the data collected at UCB, which performed the longest measurement.

30.03% of the connections contain SCTs. As in the active scans, SCTs in certificate extensions are by far the most common case (20.45% of connections), but SCTs are also commonly seen in TLS extensions, namely 9.56% of all connections, or 13.64% of connections where the client supported the extension (if a client does not advertise support, the server is not allowed to send an SCT via a TLS extension). This shows that even though SCTs are only embedded into the TLS extension by a small number of sites (1.6k out of 1.5M), these sites amount to a significant number of connections (248M of 779M). A closer inspection reveals that 56% of the domains can be attributed to Google. Some other major sites like Snapchat and Amazon also are present. This confirms the results of our active measurement that show SCTs over TLS are more commonly used by popular domains. As in the active scans SCTs in OCSP staples are rare; they are only observed in 155.8k connections. The lack of SCTs in OCSP extensions is not caused by the lack of client or server support of OCSP stapling; we saw 1.8G (70.14%) connections in which clients advertised support for OCSP stapling. Servers replied with stapled OCSP responses in 13.64% of these connections (248.1M).

	Berkeley	Munich	Sydney
Observation period	4.4.–2.5.	12.-16.5.	12.–16.5.
Total connections	2.6G	286.7M	196.2M
Connections with SCT	778.7M	72.7M	57.5M
Conns. SCT in Cert	530.4M	58.3M	43.9M
Conns. SCT in TLS	248.1M	14.4M	13.6M
Conns. SCT in OCSP	155.8k	37.6k	31.1k
Total certs	1.5M	178.7k	115.8k
Certs with Assoc. SCT	76.5k	46.9k	29k
Certs with X509 SCT	74.9k	46.6k	28.9k
Certs with TLS SCT	1.6k	299	177
Certs with OCSP SCT	20	29	10
Total IPs	962.3k	405k	226k
v4 IPs	737.2k	344k	226k
v6 IPs	225.1k	61k	N/A
IPs SCT	284.4k	116.6k	65.9k
v4 IPs SCT	222.3k	102.7k	65.9k
v6 IPs SCT	62.1k	13.9k	N/A
IPs X509 SCT	269.2k	110.1k	62.4k
IPs TLS SCT	15.3k	6.4k	3.5k
IPs OCSP SCT	518	208	43
Total SNIs	6.5M	983k	N/A
SNIs SCT	1.9M	282.9k	N/A
SNIs X509 SCT	852.1k	179.2k	N/A
SNIs TLS SCT	1.1M	103.7k	N/A
SNIs OCSP SCT	27	98	N/A

Table 4: Passive SCT data. SNIs from Sydney are not available as only outbound packets were observed (see Section 4).

Our data collection at UCB also extended to ports other than 443. 74,311 of certificates (99.2%) with an embedded SCT were encountered on port 443 (followed by 279 cases observed in TLS connections on port 80). For SCTs over TLS and OCSP the situation is similar. As expected, we did not encounter precertificates in active or passive data.

5.2 Properties of Certificates with SCTs

A small number of CAs are responsible for the majority of issued certificates containing embedded SCTs: Symantec issues 67.16% of all certificates with SCTs through their brands Geotrust (33.67%), Symantec (28.75%), and Thawte (4.74%). This is probably caused by Google requiring Symantec to log all its certificates due to previous incidents of mis-issuance [65]. Chrome shows a warning for Symantec certificates without SCTs. Other CAs who issued significant numbers of certificates containing embedded SCTs are GlobalSign (11.91%), Comodo (11.66%), and StartCom (3.19%). StartCom and its parent company WoSign are already distrusted by Mozilla for new certificates [78] and has been distrusted by Google Chrome as planned in September 2017 [76]. Although we observe a substantial number of certificates containing SCTs overall, only a few CAs are responsible for this; some of whom are required by Google to provide SCTs. This indicates that currently not many CAs seem interested in providing embedded SCTs, showing that opinions did not significantly change since 2014 [13].

Table 5 shows which logs created the SCTs that are embedded in certificates, as well as sent over TLS extensions. The most commonly used logs are currently operated by Symantec, Google, and DigiCert, although the exact percentages differ between active scans and passive observations. The Google and DigiCert logs accept certificates from a wide range of CAs; Symantec only allows certificates from a handful of CAs logs [66]. This shows that CAs are likely to submit their precertificates to a low number of logs, and an even lower number of operators, causing a certain concentration of trust. We hope that this picture will change in the future when more public log servers are established.

Table 6 shows how many SCTs from different logs and log operators are added to certificates. This is an important metric: Google's future plans are to require that certificates be included in a number of logs operated by different entities [28]. Our analysis shows that most certificates are logged by more than one log operator already. However, we encounter 16.42k cases of certificates that are logged exclusively by one operator; 16.3k of these are logged exclusively by Google. Certificates logged in one log only are rare and mostly contained in Symantec's Deneb log, which we discuss in the next section.

5.3 Invalid SCTs

We validate all SCTs. This nearly always succeeded, with a few notable exceptions.

In exactly one case, a correctly CA-signed certificate contained invalid *embedded* SCTs. The certificate, issued to *www.fhi.no* by Norwegian CA Buypass, was observed in both active and passive measurements. The three embedded SCTs are from Google Aviator, Venafi, and Symantec logs. We contacted Buypass regarding these invalid SCTs. Although previously unaware of the problem, they confirmed that a corner case in their implementation caused the embedding of SCTs belonging to a different certificate for the same domain. Buypass revoked the certificate containing the faulty SCTs [20] and issued a new certificate with valid SCTs [19].

Invalid SCTs in our data set originate primarily from TLS extensions. Sending SCTs in the TLS extension typically requires manual configuration on the server side. In our active scan, we encounter this case on 121 domains serving 101 certificates. 91 of these were issued by Let's Encrypt, who currently do not embed SCTs in their certificates. Erroneous SCTs can be caused by operators updating their server certificates manually, forgetting to update the SCT configuration.

Another typical source of invalid SCTs is Symantec's Deneb log. This is a special log: all domains in the issued certificates are truncated to the second-level domain, excluding subdomains [67]. Validation of a certificate against a Deneb log signature requires modifying the received certificate and truncating all domains contained within. The Deneb log is not trusted by Google or, to our knowledge, anyone else. We are not aware of any actual implementation of this highly unusual validation method, which sidesteps one of the most

Active SCT in Cert	Active SCT in TLS	Berkeley Passive SCT in Cert	Berkeley Passive SCT in TLS
Symantec log (81.26%)	Symantec log (62.71%)	Symantec log (79.69%)	Symantec log (96.16%)
Google 'Pilot' log (79.9%)	Google 'Rocketeer' log (58.53%)	Google 'Pilot' log (78.95%)	Google 'Pilot' log (51.51%)
Google 'Rocketeer' log (31.73%)	Google 'Pilot' log (58.42%)	Google 'Aviator' log (42.79%)	Google 'Rocketeer' log (50.19%)
DigiCert Log Server (26.95%)	Google 'Icarus' log (14.35%)	Google 'Rocketeer' log (38.35%)	WoSign ctlog (2.64%)
Google 'Aviator' log (25.68%)	Google 'Aviator' log (9.49%)	DigiCert Log Server (26.42%)	Google 'Skydiver' log (1.7%)
Google 'Skydiver' log (8.31%)	Venafi log (7.46%)	Symantec VEGA log (8.08%)	Venafi log (0.44%)
Symantec VEGA log (3.98%)	WoSign ctlog (4.63%)	Google 'Skydiver' log (7.22%)	DigiCert Log Server (0.38%)
StartCom CT log (1.49%)	DigiCert Log Server (4.07%)	Izenpe log (0.58%)	Google 'Icarus' log (0.38%)
WoSign ctlog (0.67%)	Google 'Skydiver' log (1.69%)	Venafi log (0.54%)	Google 'Aviator' log (0.25%)
Izenpe log (0.14%)	Venafi Gen2 CT log (1.58%)	StartCom CT log (0.38%)	NORDUnet Plausible (0.06%)

Table 5: Top logs by number of certificates with SCTs, for active scans from Sydney and passive monitoring in Berkeley. Numbers as percentage relative to all certificates in scan with SCT in Cert/TLS. A certificate typically has more than one SCT.

Logs				Unique Log Operators			
# Logs	Certificates (Active-All)	Certificates (Berkeley-Mon)	Connections (Berkeley-Mon)	# Ops.	Certificates (Active-All)	Certificates (Berkeley-Mon)	Connections (Berkeley-Mon)
1	156 (0.02%)	32 (0.04%)	134k (0.03%)	1	16.42k (1.89%)	734 (0.98%)	312k (0.06%)
2	601.87k (69.37%)	33.76k (45.05%)	145M (27.36%)	2	740.99k (85.4%)	67.55k (90.14%)	453M (85.47%)
3	107.63k (12.4%)	25.22k (33.65%)	165M (31.24%)	3	110.19k (12.7%)	6.65k (8.88%)	76M (14.47%)
4	57.24k (6.6%)	10.54k (14.06%)	217M (40.99%)	4	42 (0%)	2 (0%)	37 (0.00%)
5	100.73k (11.61%)	5.39k (7.19%)	2.0M (0.38%)	5	0 (0%)	0 (0%)	0 (0.00%)

Table 6: Number of logs/log operators in certificates. Percentage relative to all certificates/all connections with SCT.

important properties of CT—to make a list of all issued certificates per domain available to everyone. Symantec advertises this option for customers who do not want to disclose the existence of their domains to monitors. In our active scans, we encountered 129 certificates logged in Deneb logs. Interestingly, we found that 87 of them were also included in Google logs and 2 in a Comodo log, defeating the purpose of Symantec's Deneb log in such cases. Of these, 64 certificates were issued for Amazon domains.

In our passive data from UCB, we find a particularly interesting type of certificate with invalid SCTs not present in our active scan. We observe 425 cases in which the certificate contains an extension with an SCT object identifier, but no SCT data. Instead, the extension contains the string 'Random string goes here'. The certificates cover 5 domains, with 352 certificates issued for **.cloudfront.com* and 68 for *twitter.com*. None of the certificates could be validated against the CA certificate mentioned as the issuer. The server IP addresses that served the certificates revealed several hosting and dedicated server providers like LogicWeb. Manually trying to connect to a number of these IP addresses resulted in a TLS handshake error; no certificate was sent. Examining a subset of the certificates reveals that they are exact clones of existing certificates: the serial number, subject, issuer and all extensions except for the SCT are exactly the same as in the real certificates. We can only speculate about the use of these certificates. They could be used to disguise traffic as belonging to popular services. A cursory inspection would show normal-looking TLS connections with certificates seemingly belonging to Twitter or CloudFront.

5.4 CT Inclusion Status

A natural question to ask is whether logs are well-behaved and include every certificate for which we find a valid embedded SCT. Inclusion of normal certificates is straightforward to check, but precertificates require reconstruction from their final end host certificate. We enhance a preexisting X.509 certificate parsing library for PostgreSQL, adding functionality for stripping the signature, SCTs, and poison extension from certificates. There also is a second, more complicated method for precertificates where the precertificate chains to a special precertificate CA certificate, not that of the final issuing CA. For these exotic cases (a handful of certificates in our scan), we manually verified inclusion in the respective CT logs. In conclusion, *all* encountered certificates with a valid embedded SCT were correctly logged by the respective CT logs. This is a strong indication that CT's precertificate system works almost flawlessly, with the one negligible exception of the *www.fni.no* certificate.

6 HSTS AND HPKP

In our active scans, we probe servers for HSTS and HPKP headers by sending HTTP *HEAD* requests. We analyze deployment, consistency, lifetime, and cryptographic validity of the received headers. Table 7 provides the counts of domains that respond with HTTP 200 ('OK') to our request. We collapse responses from the same domain if the HSTS and HPKP headers are consistent across all IP addresses. We combine the analysis of HSTS and HPKP due to the similarity of some of their attributes.

	HTTP 200	HSTS	HPKP
MUC IPv4	26.8M	960.0k (3.59 %)	5.9k (0.02 %)
SYD IPv4	26.5M	948.5k (3.58 %)	5.8k (0.02 %)
MUC IPv6	1.2M	38.8k (3.36 %)	1.0k (0.09 %)
Total	27.8M	1.0M (3.60 %)	6.2k (0.02 %)
Consistent	27.8M	984.1k (3.54 %)	6.2k (0.02 %)

Table 7: Unique HTTP code 200, HSTS, and HPKP domains responding to MUCv4, SYDv4, SYDv6, and any scan. Last row displays domains with consistent headers across scans.

6.1 Header Consistency

We first investigate intra-scan consistency, *i.e.,* whether the headers for a domain are consistent within each of the TUMv4, TUMv6, and SYDv4 measurements. For each scan, we find a tiny fraction (≈0.003%) of domains exhibiting inconsistent settings, largely (>65% of cases) caused by domains setting HSTS or HPKP headers on one set of IP addresses, but not on another. Many of the remaining cases set inconsistent HSTS *max-age*, or *includeSubDomains* values; one domain pins different HPKP keys. The inconsistent domains within individual scans are 6 for MUCv6, 25 for MUCv4, and 22 for SYDv4, a tiny fraction compared to the millions of scanned domains per vantage point. This group of inconsistent domains partially consists of services that are globally distributed under shared administration, such as pooled NTP and OpenPGP Key Servers. We limit the following analyses to domains consistent within one scan.

We evaluate inter-scan consistency, *i.e.,* whether headers are consistent between scans. We find about 2% of HSTS/HPKP-enabled domains to serve different headers across at least two scans; the difference is nearly exclusively caused by configurations serving HSTS in one scan, but not the other. In detail, we find 15k domains inconsistent between the MUCv4 and SYDv4 scan, and 754 domains inconsistent between the MUCv4 and MUCv6 scan. From sample analysis, we identify three potential reasons for this behavior: (a) timing differences between our scans (b) differently configured IP anycast services, revealed using speed-of-light constraints [63] (c) load-balancers with inconsistently configured servers. We limit the following analyses to domains that serve consistent headers across all scans.

6.2 Deployment

3.5% (984k) of the domains with consistent and HTTP-200 headers support HSTS. .2% of HSTS-enabled domains send incorrect HSTS headers—typically due to typographical mistakes, such as *includeSubDomains* missing the plural *s*. 41k domains do not use HSTS effectively, setting the *max-age* attribute to 0, resulting in a 'deregistration' from HSTS use (24k domains), to a non-numerical value (16k domains), or to an empty value (1k domains). For HPKP, we find that only 6181 of all 28M domains (0.02%) send the header. Of these, 29 do not send a valid *max-age* directive and 12 do not contain any pins.

Max-Age: The *max-age* attribute indicates the lifetime of HSTS and HPKP headers, which browsers will update on every domain visit. Figure 2 shows the distribution of *max-age* across all HSTS domains, HPKP's *max-age* for the subset of domains that also support HSTS ($\text{HPKP}|_{\text{HSTS}}$), as well as HSTS's *max-age* for the subset of domains that also support HPKP ($\text{HSTS}|_{\text{HPKP}}$). The intersecting sets generally have shorter durations with the majority of HPKP *max-age* values being 10 minutes (33%), 30 days (22%), and 60 days (15%). HSTS domains that also send HPKP headers choose 5 minutes (32%), 1 year (26%), and 2 years (14%). The largest values are sent by the set of all HSTS domains with 2 years (46%), 1 year (32%), and 6 months (10%). The median *max-age* of HSTS is one year, but only one month for HPKP. This suggests operators exercise caution when using HPKP, which carries high availability risk through lock-out (cf. the Cryptocat lock-out [69]). We also note an extreme outlier setting an HSTS *max-age* of 49 million years (a likely accidental duplication of the string for half a year).

includeSubDomains: 56% of HSTS and 38% of HPKP domains use the attribute *includeSubDomains*. This attribute enables HSTS for all subdomains of the domain setting this attribute. This has many benefits, for example, helping to avoid insecure cookies from subdomains. However, it may cause operational difficulties when subdomains do not actually support HTTPS. As some domains do not even use subdomains, it is difficult to assess this percentage.

Preloading Lists: We investigate the HSTS *preload* list included in Chrome [14], which also is the base for Mozilla's preloading list [48]. A domain can be added to the list by (a) setting the HSTS directive, (b) including the non-RFC *preload* parameter and (c) opting in through sites such as Chromium's hstspreload.org. Interestingly, we find a large fraction (379k, 38%) of scanned domains to include the *preload* directive, but only 23k domains in the preload list of the current version of Chrome (58), with the intersection consisting of just 6k domains. Two possible explanations are that the inclusion

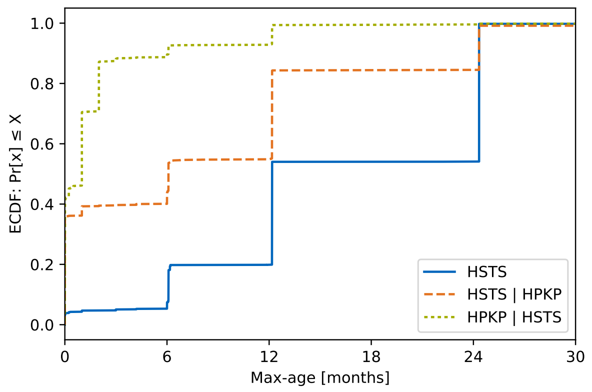

Figure 2: Distribution of the *max-age* attribute for HPKP and HSTS headers: Domain owners typically set much higher max-ages for HSTS than HPKP.

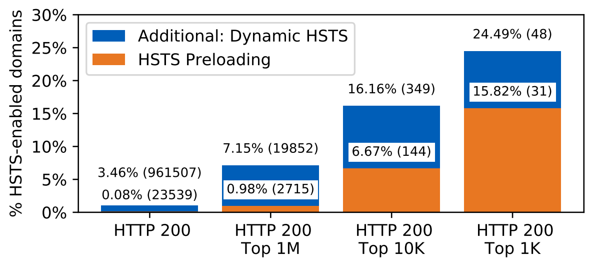

Figure 3: HSTS: Significant usage among top domains. Preloading essentially absent in general population, but with significant deployment among top domains.

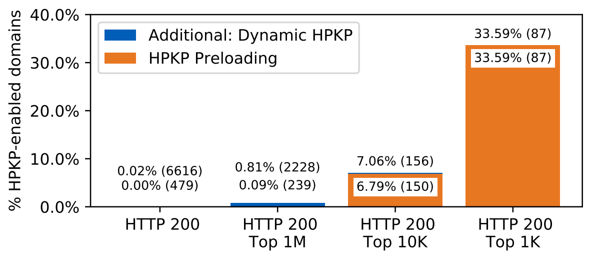

Figure 4: HPKP: Low usage among general population, significantly higher usage through preloading list among Alexa top domains.

process is slow to catch up, or operators do not follow all prescribed steps for inclusion, e.g., because they just copy directives from tutorials.

Our scans include 13k of the 23k domains in the preload list (the remainder are domains without A/AAAA records, from TLDs not in our list, or subdomains we do not scan). We successfully connect to 6.6k of these 13k domains. 6026 of them send the HSTS header and 5656 include the *preload* attribute. The remaining domains do not satisfy the preloading criteria anymore and will be removed from the preloading list eventually. Further examination of HSTS-preloaded domains reveals that some popular domains only preload subdomains, but not their base domain: for the Alexa Top 1M list, 91 of the 2715 preloaded domains only preload a subdomain. One example is *theguardian.com*, which sets dynamic and preloaded HSTS for its *www* subdomain, but not for its main domain. This exposes users of the base domain to HTTPS stripping and redirect attacks. Another example is Google who enables HSTS for select subdomains. We contacted Google who stated that each service and its subdomain has to be verified individually, making the HSTS roll-out a long process.

There is no publicly accessible preloading mechanism for HPKP. Browser vendors, however, include important domains in their internal HPKP preloading list. Mozilla's HPKP preloading list, which extends Chrome's list [47, 48], includes a total of 479 domains, mostly from Google, Facebook, Yahoo, Twitter, Mozilla, and the Tor project.

Public Key Pinning: We analyze the validity of HPKP pins. The majority (86.0%) of scanned HPKP domains use HPKP correctly and provide at least one valid pin. Examining non-matching cases reveals that for 8.5% of HPKP domains the certificate is known to us, but missing from the handshake. Exploring the top 5 cases reveals 4 intermediate CA certificates missing from the handshake (a TLS standard violation, but accepted by browsers) and one certificate falsely copied from an HPKP tutorial webpage.

The majority of the remaining 5.5% of HPKP domains with pins where we find no matching public key in our certificate set use bogus pins, many being syntactically invalid SHA256 hashes. The top 3 are the pins from the RFC example section, the text *<Subject Public Key Information (SPKI)>*, and *base64+primary==, base64+backup==*. Pins that do not have the correct format are ignored by browsers.

Deployment Ranking: Figures 3 and 4 differentiate HSTS and HPKP usage for both dynamic and preloaded deployment across domain rank. Note that our 100% baseline is the fraction of Top 1M, Top 10k, and Top 1k domains answering with HTTP 200, amended by all preloaded domains. For both technologies, we find that few domains of the base population deploy them. The rising share of dynamic and preloaded domains with domain popularity is encouraging. This is in line with expectations that more popular domains also have more resources to configure and maintain these security extensions. The share of preloading among top domains, especially for HPKP, also is encouraging.

Comparison to Related Work: We compare our results to those by Kranch and Bonneau [42], who evaluated HSTS and HPKP extensively in 2014. Both technologies have gained much usage, in both dynamic and preloaded fashion: The HSTS preload list has grown from 1258 domains in 2014 to 23.5k domains in 2017, and the number of dynamic HSTS domains in the Alexa Top 1M list has grown from 12.5k in 2014 to a lower bound of 18k in 2017. As HPKP was still in the standardization process in 2015, Kranch and Bonneau found only 18 domains supporting HPKP. This number has risen to about 6k. We can confirm many of the issues and oddities observed by Kranch and Bonneau, such as mistyped directives, mismatches between preloaded and dynamic HSTS domains, and redirections from top domains not covered under HSTS. We agree with their conclusion that forcing all operators to determine and configure a *max-age* is prone to mistakes, and support their suggestion of a reasonable default setting.

7 SCSV DOWNGRADE PREVENTION

We evaluate SCSV downgrade prevention support [50]. SCSV is a pseudo ciphersuite value sent by the client when retrying to connect to a server with a lower TLS version after a first attempt failed (see Section 2). We distinguish between four outcomes. First, the server correctly aborts connections sending the SCSV using an alert or some other message. Second, the connection fails due to a transient error (*e.g.,* timeout). Third, the server incorrectly continues with the connection. Fourth, the server incorrectly tries to continue the connection, but chooses parameters not supported by our

Scan	Conns.	Fail.	Domains	Incons.	Abort.	Cont.
MUCv4	55.68M	5.4%	48.41M	.1%	96.2%	3.8%
SYDv4	57.95M	4.7%	48.75M	.1%	96.1%	3.8%
MUCv6	5.11M	.6%	3.27M	.0006%	99.5%	.5%
Merged[1]	N/A	N/A	51.16M	.008%	96.3%	3.7%

1: *Merged* dataset exclusively contains per-scan consistent domains.

Table 8: SCSV statistics from active scans: >96% of domains correctly abort TLS connections.

local TLS client. This last case affects only .03% of domains, which we count as attempts to continue the connection.

Table 8 displays results for our active scan both per scan and for merged scans. The results show that >96% of HTTPS-responsive domains implement RFC 7507, despite it being relatively recent (April 2015). The Alexa Top 1M, 10k, and 1k domains have equally high coverage.

This high percentage of support may stem from the inclusion in cryptographic libraries. Server operators do not have to change any settings, but just use a recent cryptographic library, making deployment very easy. 5 of the 7 domains in the Alexa Top 100 list that do not support SCSV are from Microsoft and use IIS according to HTTP headers. While there is no official statement from Microsoft regarding SCSV support in IIS and SChannel, several blog posts [34, 45] indicate that SCSV support has been missing for a while. Given the 11% market share of IIS among HTTP servers [75], it seems reasonable that a large fraction of the non-supporting domains may be caused by the lack of SCSV support in IIS/SChannel.

We also evaluate whether SCSV is used in user-initiated connections by investigating our passive data. We count source and destination IP address tuples that use the SCSV pseudo-cipher at least once, and find 126k (0.2%) in Germany and 198k (0.1%) in the USA. This indicates that SCSV protection ciphers are used in the wild across many clients and servers.

8 DNS-BASED SYSTEMS

We investigate the use of CAA and DANE-TLSA (for TCP port 443) records for domains that yielded an A or AAAA record in our DNS resolution. As described in Section 2, CAA and DANE-TLSA are DNS records that help with certificate issuance and verification, respectively. We perform scans from Munich and Sydney roughly two weeks after our TLS scans. Even with this time difference, the majority of lookups was successful: less than 100 attempts at resolving records failed where previous attempts (A, AAAA) had succeeded.

Table 9 presents an overview of domains with at least one CAA/TLSA record and details if they are DNSSEC verifiable. Note that domains may have multiple CAA and TLSA records. TLSA mandates DNSSEC, CAA does not. Given the low rate of DNSSEC deployment [73], it is a good sign that 20-25% percent of CAA and around 75% of TLSA records use DNSSEC. While CAA/TLSA records differ slightly between Munich and Sydney, we do not encounter any divergence in DNSSEC verification. Slight differences are expected due to timing variation and nameservers using IP anycast to redirect to closer endpoints.

	SYD	MUC	Intersection	Top 1M
CAA	3243 (100%)	3509 (100%)	3057 (100%)	340 (100%)
signed	674 (21%)	899 (26%)	621 (20%)	53 (16%)
TLSA	1697 (100%)	1364 (100%)	1246 (100%)	100 (100%)
signed	1330 (78%)	1042 (76%)	973 (78%)	89 (89%)

Table 9: Number of domains with CAA and TLSA records, with and without validating DNSSEC signatures.

CAA has a larger deployment (340) in the Top 1M compared to TLSA (100). However, there are more signed TLSA domains (89 compared to 53). The preference among the higher ranked domains may indicate that the record is viewed as beneficial.

CAA: A CAA record basically consists of one or more *properties*, which are key-value pairs. The *issue* property is used to specify which CA may issue certificates; *issuewild* is the same, but for wildcard domain certificates. The *iodef* property provides means to contact a domain owner in a standardized (machine-readable) way, *e.g.,* when a CA receives a request to issue a certificate that violates the CAA policy.

We first analyze the top CAs for the *issue* property. It is important to note there is currently no agreed mapping of human-readable strings to CAs. The results are consistent from both vantage points, hence we report values from our Sydney scans.

From the vantage point in Sydney, we find a total of 3,834 records containing the *issue* property. Unsurprisingly, most records are contributed by domains in the *.com* zone (1,742)—however, the second most common zone is *.de* (682), well ahead of the similarly-sized generic *.org* and *.net* (463 and 447, respectively). No other zone contributes more than 75 records.

The most common string is *letsencrypt.org*, in 2,270 records. Let's Encrypt launched in 2016, is free and fully automated. Domain ownership is verified and certificates are issued using the ACME protocol [9]. The next four strings, *comodoca.com*, *symantec.com*, *digicert.com*, and the Google CA *pki.goog* all appear with similar numbers (246, 233, 195, and 195 respectively). However, Comodo also appears with the string *comodo.com*, and brands owned by other CAs appear with their own string (*e.g.,* GeoTrust, owned by Symantec, or RapidSSL, owned by Comodo). We find a total of 55 combinations and spellings. In 63 cases, operators set a semicolon; this signifies that no CA is allowed to issue certificates.

We find that the *issuewild* property is used less often than *issue*: only 1,088 such records exist (for 1,064 domains). They show an entirely different use pattern. 756 records are set to a semicolon, meaning that no CA may issue wildcard certificates. We find a few dozen domains that choose different settings for *issue* and *issuewild*. In the vast majority of these cases, *issue* is set to Let's Encrypt (which does not yet issue wildcard certificates) and *issuewild* to another mainstream CA.

Finally, we find 1145 *iodef* records, across 1,141 domains. The standard allows email and HTTP URLs for reporting. About 220 records violate this requirement; manual inspection shows that all are meant to be email addresses (mostly a missing *mailto:*). 908 domains use an email and just 13 an HTTP(S) URL. We find a total of 380 unique email addresses in the records. By attempting repeated SMTP dialogues and testing for responses for *RCPT TO:*, we find that only 63% of *iodef* email addresses actually exist.

We find only 9 unique HTTP(S) URLs. According to RFC 5070, a web service should reply with a 204 response to a POST with an empty key-value pair. In our tests, this was only the case for 2 URLs; 5 returned an error and 2 served normal websites.

In conclusion, it seems that domain owners are conscious of security when choosing their CAA records. They mostly disallow wildcarded certificates, and they show a very clear preference for Let's Encrypt.

TLSA: TLSA is a DNS-based form of certificate pinning. TLSA has been well studied; the SecSpider dashboard even tracks it for DNSSEC-secured zones [73, 82].

While SecSpider [73] and our domain sets are not congruent, SecSpider's authors report 2306 base domains with TLSA enabled for HTTPS as of August 23, 2017. We find this very low deployment in line with our 1697 base domains.

We next investigate the 4 pinning ("certificate usage") types provided by TLSA: The first two cases require the entire certificate chain to pass validation (via the root store): in type 0, a root or intermediate certificate in the certificate chain is pinned; in type 1 it is the end-entity certificate. The other two pinning types support bypassing root store validation: in type 2, a new trust anchor (root certificate) that must be used in the validation is pinned. Type 3, pinning to an end-entity certificate outside any certificate chain, can be used, for example, to pin self-signed certificates.

The findings from SecSpider on September 28, 2017, are roughly in line with our results. Both us and SecSpider find minor deployment for PKI-chained type 0 (SecSpider 1% vs. our 2%) and type 1 (10% vs. 7%). Records of type 3 are the clear majority (90% vs. 79%). For type 2, we find 11%. SecSpider did not offer data for type 2; this must be taken into account when comparing the above relative numbers. Both scans support the conclusion that pinning a self-signed or otherwise non-verifiable certificate seems to be the primary use case for TLSA. Apart from numbers for type 2, the slight differences can also stem from a different set of scanned domains, different counting (zones vs. base domains), and a slight timing offset between measurements.

Comparison with Related Work: Szalachowski and Perrig [68] investigated the use of CAA and TLSA among the Alexa Top 100k domains in August 2016.

They find 15 CAA records, which have grown to 102 in our April 2017 scans. We re-scan on September 4, 2017, and find 216 CAA records on Alexa Top 100k base domains. Given the CAA record's youth and its September 2017 effectiveness, we consider this growth in magnitude plausible.

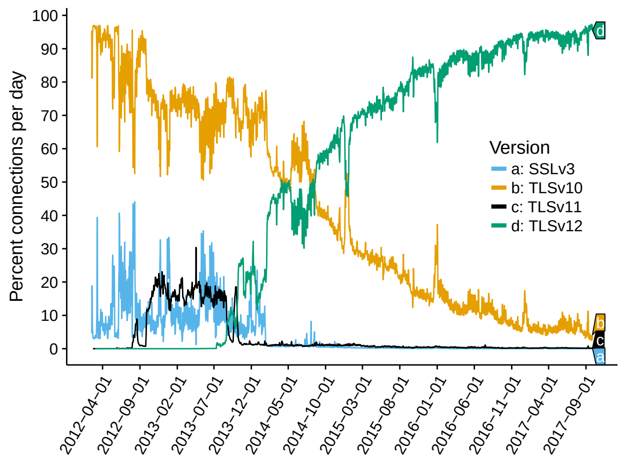

Figure 5: Ratio of SSL/TLS versions seen in established connections. SSL 2 and TLS 1.3 did not see significant use and were excluded from the graph.

Szalachowski and Perrig report 48 TLSA records for base and *www* subdomain, in contrast to our 18 domains with TLSA records on the base domain in April 2017. We rescan on September 4, 2017, and find 36 Alexa Top 100k base domains and 35 *www* subdomains with a TLSA record, totaling in 71 records. This indicates that a large fraction of a TLSA record count can come from the *www* subdomain, making these numbers plausible and indicating slow TLSA growth.

9 TLS VERSION ADOPTION OVER TIME

So far, we discussed the current use of recent security additions to the HTTPS ecosystem. In this section, we take a step back to give a brief overview of TLS version adoption in the last 5 years. We use data from the ICSI Notary [8], a large-scale passive monitoring effort of the TLS ecosystem operating since February 2012. To date, it has observed more than 240.1G connections (221.9G on port 443) containing 14.6M unique certificates.

Figure 5 shows the SSL/TLS versions negotiated in sessions. Even in 2013, there was still significant use of SSL 3 (draft published 1996). TLS 1.0 (standardized 1996) was the most commonly used version of TLS when the Notary was established, and this remained true until the end of 2014. TLS 1.1 (standardized 2006), and TLS 1.2 (standardized 2008) only gained traction years after their standardization finished. While TLS 1.1 saw some increased use in 2013, it never gained significant adoption. Our data suggests that most sites directly changed from TLS 1.0 to libraries supporting TLS 1.2. This is supported by the fact that Unix/Linux based servers will often use OpenSSL, which introduced support for TLS 1.1 and TLS 1.2 simultaneously in the release of OpenSSL 1.0.1 on March 14, 2012.

In preparation for this paper, we implemented support for parsing TLS 1.3 sessions in Bro. This was added to Bro 2.5 (released in Nov. 2016), with some sites running beta-versions of the code before the release. To date, we encountered more

Y↓ , X→	SCSV	CT	HSTS	HPKP	CAA	TLSA	Top 1M	HTTP 200
n	26M	2.3M	944k	6k	1.2k	0.5k	0.3k	28M
SCSV	100.00	95.65	67.86	96.03	92.57	91.49	96.08	94.94
CT	8.37	100.00	7.10	45.88	12.14	13.54	13.36	8.31
HSTS	2.43	2.90	100.00	92.21	49.12	70.21	6.60	3.40
HPKP	0.02	0.12	0.61	100.00	9.82	19.92	0.72	0.02
CAA	0.00	0.01	0.07	1.98	100.00	14.70	0.04	0.00
TLSA	0.00	0.00	0.04	1.66	6.07	100.00	0.02	0.00
Top 1M	1.01	1.60	1.93	32.12	8.63	8.32	100.00	0.99
HTTP 200	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00

Table 10: $P\left(Y|X\right)$ in %, giving the empirical probability that technology Y is deployed when X is. For comparability across all features, sets only contain HTTP 200 domains, making these numbers incompatible to in-depth analysis per feature. Highlighted cells are discussed in Section 10.1.

than 7 million connections negotiating different TLS 1.3 drafts. The number of these connections peaked during February 2017 with up to 36k successful connections per day, when Google enabled TLS 1.3 support by default in Chrome 56. These numbers decreased to 1k to 5k successful connections per day after Google disabled support again due to compatibility problems [59]. We see the early use of TLS 1.3 as a sign that the community is more security sensitive than it was in 2012, when our measurement began. We will continue monitoring the development and deployment of TLS 1.3.

10 DISCUSSION

In this section, we discuss the overall state of HTTPS security based on our measurements. First, we map protection mechanisms against attack vectors and assess the number of protected domains. Next, we relate our findings to the deployment effort and risks to site availability. Furthermore, we give specific advice to different stakeholders how to improve the deployment of security extensions. Finally, we discuss the value of employing multiple vantage points and IP protocol versions to assess overall security.

10.1 Correlation of Security Feature Application

In this section, we investigate the correlation between deployment of different features. Table 10 shows the conditional probability for a feature Y to be effectively deployed given that another feature X is effectively deployed.

The lower left triangle of the matrix shows that deployment of a frequently deployed feature such as SCSV or CT does not imply the use of less common features. The upper right triangle offers more interesting insights, of which we discuss the highlighted cells here:

First, effective deployment of SCSV is less frequent for domains that use HSTS. Further investigation of this intriguing fact reveals that 280k domains, roughly equal to the drop-off compared to average deployment are hosted by the controversial [77] provider Network Solutions/web.com. The hoster apparently enabled HSTS for a large set of domains, without support for SCSV and even does not provide valid certificates for those domains. We conclude from this that the drop-off in support of SCSV for domains that use HSTS ($SCSV|HSTS$) compared to the overall SCSV population stems from this large hosting provider who handles SCSV incorrectly and sets HSTS for a large population of likely unused domains.

Attack Vectors	SCSV	CT	HSTS	HPKP or TLSA	HPKP	HPKP PL.	HSTS PL.	CAA	TLSA
TLS Downgrade	●								
TLS Stripping			○				●		
MITM w/ fake Cert				○	○	●			●
Mis-Issuance Detection		●							
Mis-Issuance Prevention								●	
Domains Protected	**49.2M**	**7.0M**	**0.9M**	**7485**	6616	479	27759	3057	973
Intersection[1]	49.2M	6.1M	**67,153**	2879	2827	—	—	24	**2**
Top 10k →	6789	1959	349	158	156	150	144	20	3
Intersection[1]	6789	1799	85	6	6	—	—	0	0

1: Number of domains that deploy the intersection of mechanisms, from left to right.

Table 11: Attack vectors, protection mechanisms, and empirical coverage, sorted by Top 10K deployment. ○ denotes partial protection (susceptible to Trust On First Use attacks), and ● denotes full protection.

Second, domains that use HPKP very frequently also use CT and HSTS headers. We expect users who successfully master the complicated HPKP setup to also deploy other techniques. One reason why CAA and TLSA usage remains relatively low among HPKP users, could be the required control over the domain's DNS server.

Third, use of CAA or TLSA is frequently combined, and often correlates to HSTS or HPKP deployment. Given the low dissemination of CAA and TLSA, it is not surprising to find its users be aware of other, more common security techniques.

10.2 Protection Against Attack Vectors

Table 11 shows which HTTPS security extensions protect against specific attack vectors. This is based on work by Clark and van Oorschot [18], who in 2013 theoretically evaluated various HTTPS security extensions. We contribute empirical evidence about their use.

The upper section of Table 11 maps attack vectors to protection mechanisms. Most protection mechanisms defend against exactly one attack vector. Only HPKP and TLSA overlap, *i.e.,* both protect against MITM attacks by offering key pinning. In general, however, multiple protection mechanisms must be combined to protect against all attack vectors. We thus analyze how many domains are protected with multiple mechanisms. This is shown in the lower sections of Table 11. We start on the left with the most common protection mechanisms, and then successively intersect the set of protected domains with the set of domains that are protected with the mechanism(s) to the immediate right. Note that we exclude preloading lists from the intersection as they are only one *option* to provide HSTS and HPKP.

We see the number of protected domains drop one order of magnitude for each of the first four mechanisms. We also find a large drop in HSTS protected domains when intersecting with SCSV and CT. In fact, only about 7% of all HSTS domains are also protected with SCSV and CT. We, therefore, deduce that most hosts only deploy one or two protection mechanisms. This is also the case for mechanisms which require minimal configuration effort and carry low risk of accidentally making a site unavailable. Ultimately, we find only 2 domains (*sandwich.net* and *dubrovskiy.net*) to deploy all security mechanisms investigated in this work. Unfortunately, the latter uses the StartSSL CA, which is now distrusted by Chrome.

10.3 Top 10 Validation

We validate our findings for the Alexa Top 10 domains using SSL Labs [58]. Table 12 shows the settings for the Alexa Top 10 domains. In line with our overall results, we find almost universal support for SCSV, but only few domains deploy CT, HSTS, or HPKP. Only *google.com* uses CAA and no Top 10 domain deploys TLSA. We emphasize the high and correct usage of SCTs as a TLS extension, caused by Google domains. Please note that our investigation only looks at base domains—redirects are not followed. As noted by Kranch and Bonneau [42], the security extensions covered in this work are required to work on the base domain, as any insecure redirect can be exploited by attackers. Under this precept, we do not attribute HSTS to domains that only use HSTS on subdomains, such as in the case of *google.com* where HSTS is supported for a lot of subdomains such as *www.google.com*, but not the base domain (see section 6.2).

10.4 Correlating Effort, Risk, and Usage

We relate deployment effort, risk to site availability, and measure deployment of technologies that have emerged after the DigiNotar incident in Table 13.

We classify effort as follows: *None*, where the server administrator has to take no action (*e.g.,* SCSV, embedded SCTs). *Low* applies where the operator has to enable an extension, but no complex configuration is needed. *E.g.*, this applies to HSTS, as one can just copy a configuration string from a tutorial. *Medium* applies where the operator needs to make simple adjustments to instructions from manuals, *e.g.,* replacing the domain name or CA name, *e.g.,* for CAA. *High* effort is assigned where configuration requires careful thought or multiple steps, *e.g.,* for HPKP.

We classify availability risk as how easily misconfigurations can lead to serious availability issues for a domain. We assign *none* where no risk exists, and *low* to technologies that do not offer much risk potential or are easy and quick to fix. *Medium* applies to mechanisms that are more difficult to fix when deployed incorrectly and can affect a large user base at once. *High* is used for technologies that can harm availability for a large user base and that are difficult and slow to remediate. Examples are wrongly preloaded HPKP pins [69] and hostile pinning, where an MITM attacker sends incorrect pins to restrict user access [25].

Using this classification system, we see that technologies with low effort and low risk to availability seem to hold a bigger market share.

10.5 Improving Deployment

Our data suggests that security extensions with low (or no) deployment effort and/or little risk to availability can quickly gain market traction. This would support several hypotheses—among them, administrators shying away from deployment due to perceived effort or risk, or administrators not caring enough about additional security, or possibly being unaware of the new mechanisms. In either of these cases, it would be essential to design new extensions with a view towards minimal effort on the side of administrators. Changes in how updates are shipped can help: Enabling certain mechanisms by default, as done with SCSV in OpenSSL, could also be extended to TLS enabled web servers by sending an HSTS header by default; this approach is complementary to efforts like Let's Encrypt which can be automated to ensure a server will always serve a valid certificate. Additionally, web server software could facilitate successful deployment, *e.g.,* by providing tools to generate the correct HPKP configuration directive to pin the currently used TLS key.

Our data suggests how pronounced the effect of big players pushing the deployment of security extensions can be. Browser vendors, for example, can drive the ecosystem towards the adoption of extensions, *e.g.,* as Google has demonstrated with CT. This is most effective when their interests happen to align with those of the userbase, which is a non-technical issue.

Domain	SCSV	CT	HSTS	HPKP	CAA	TLSA
1 google.com	✓	TLS	✗	Preloaded	✓	✗
2 facebook.com	✓	X.509	Preloaded	Preloaded	✗	✗
3 baidu.com	✓	X.509	✗	✗	✗	✗
4 wikipedia.org	✓	✗	Preloaded	✗	✗	✗
5 yahoo.com	✓	✗	✗	✗	✗	✗
6 reddit.com	✓	✗	Preloaded	✗	✗	✗
7 google.co.in	✓	TLS	✗	Preloaded	✗	✗
8 qq.com[1]	✗	✗	✗	✗	✗	✗
9 taobao.com	✓	✗	✗	✗	✗	✗
10 youtube.com	✓	TLS	✗	Preloaded	✗	✗

1: No HTTPS support

Table 12: Support of investigated techniques for the Alexa Top 10 *base* domains as of April 2, 2017.

Mechanism	Standard-ized	Deployment Overall	Top 10K↓	Effort	Availability Risk
SCSV	2015	49.2M	6789	**none**	**low**
CT-x509	2013	7.0M	1788	**none**[2]	**none**
HSTS	2012	0.9M	349	**low**	**low**
CT-TLS	2013	27,759	171	**high**	**none**
HPKP	2015	6616	156	**high**	**high**
HPKP PL.	2012[1]	479	150	**high**	**high**
HSTS PL.	2012[1]	23,539	144	**medium**	**medium**
CAA	2013	3057	20	**medium**	**low**
TLSA	2012	973	3	**high**	**medium**
CT-OCSP	2013	191	0	**low**	**none**

1: Preloading list first added to Chrome in 2012

2: Requires deployment effort on CA side and a new site certificate.

Table 13: Correlation of Age, #Deploying Domains, Effort and Availability Risk of various HTTPS ecosystem security extensions, sorted by Top 10K domains. Low effort and low availability risk can drive wide-spread use.

10.6 Multi-Site and Multi-Protocol Scans

For this work we conduct a multitude of active and passive measurements, covering three continents and both IPv6 and

IPv4. To the best of our knowledge, this is the first work with such a diverse coverage. We examine how much additional information can be learned from performing such a varied array of scans, vs. only using more limited sources of data.

For *active scans*, we find the IPv4 results from Sydney and Munich to be very much in line. While we encounter some expected differences between the vantage points (like DNS servers returning different A records), these differences are very limited in scope: Only 15k out of 25M domains (.06%) serve inconsistent HSTS or HPKP headers.

The same is true when comparing our IPv6 and IPV4 scans: while there are, as expected, fewer responses for IPv6, domains that respond on both IPv4 and IPv6 are typically configured consistently: Only 754 out of 1.09M dual-stacked domains (.06% as well) serve different HSTS/HPKP headers.

Our conclusion is that multiple scans are useful for result validation, but are only required for studies that seek in-depth knowledge about how the HTTPS ecosystem presents itself from different view points. Please note that our analysis only compares HSTS and HPKP headers across domains — hosts may still serve different content, present different certificates, or exhibit different cryptographic algorithms, which were out of scope in this work. These properties, along with a detailed analysis of inconsistent domains [52], can be further analyzed based on our shared raw data.

Even for *passive monitoring*, the measurements from our three vantage points yield similar results. Again, having several vantage points available still provides valuable validation. Contrasting active and passive monitoring, we find our active scans to yield more data points and, in some respects, to allow for more in-depth analysis and more precise statements. Some of our measurements are impossible using passive data: *e.g.,* HTTP headers are not visible in passive monitoring of HTTPS. We emphasize, however, that passive monitoring remains a valuable, complementary addition. It allows to track uptake of new technologies by users, and we found in our work that it is more likely to lead to discovery of oddities, such as our discovery of connections to servers that imitate TLS handshakes to well-known sites (see Section 5.3).

10.7 Limitations and Future Work

Our work takes some important design decisions on the evaluation of domain-based security.

The first is to only look at domains that behave consistently across different target and source IP addresses and protocols. While very few domains fail this test, closer investigation of inconsistent domains may reveal interesting insight.

The second is to only look at the base domain, i.e., not prepending the *www* prefix, not following redirects, and only evaluating headers of HTTP 200 domains. While it is critical for domains to protect their base domain, which is the user's typical entry point, the correctness and behavior of redirects into subdomains is another complex field of interesting study.

The third is the more systematic investigation of "parked" domains, which, *e.g.,* use the same invalid certificate for millions of hosted domains. Carefully limiting the influence of such clusters may further sharpen domain-based analysis.

10.8 Reproducible Research & Data Release

We aim for repeatable, replicable, and reproducible research as defined by ACM [1, 64]. We publish all active scan data (traces and/or result data), as well as the source code for the utilities we used and created for this work. We can not provide more detailed information about passively captured data for ethical and legal reasons. Hosting is provided by the TUM library for long-term availability at https://mediatum.ub.tum.de/1377982

11 SUMMARY

A number of new security measures for the HTTPS ecosystem have been developed in the five years since the compromise of DigiNotar. While these techniques protect against a wealth of different attacks and would have been able to prevent or at least lessen the impact of the DigiNotar compromise, we find that deployment is disappointing for most of them. Our findings suggest a correlation between configuration effort, incurred risk to site availability, and actual deployment status. Technologies that are easy to deploy and have little risk to availability have the highest deployment (Certificate Transparency and SCSV). Those that have either high deployment effort or carry a high risk of misconfiguration have often low deployment. We note that Certificate Transparency may be a special case in our study: its deployment was massively supported by a major corporation (Google) whose business is heavily web-based and which also develops a very popular browser, hence allowing it to provide both client- and server-side support.

Our findings would support the hypothesis that operators consciously decide against deployment based on perceived effort and/or incurred risk; but it would also be consistent with a large faction of operators not caring enough about additional security to invest the effort or simply being unaware of available defenses for their sites. Empirical measurement finds its limits here: a qualitative, interview-based follow-up study may be advisable and reveal deeper insights why deployment is lacking. As such, the mission to achieve HTTPS ecosystem security is certainly not yet accomplished.

Acknowledgments:
This work was partially funded by the Major Equipment and Early Career Researcher grant schemes of the Faculty of Engineering & Information Technology, The University of Sydney; by the German Federal Ministry of Education and Research under project X-Check, grant 16KIS0530, and project DecADe, grant 16KIS0538; by the National Science Foundation under grant numbers CNS-1528156 and ACI-1348077. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of The University of Sydney, the German Federal Ministry of Education and Research or the NSF.

We thank our shepherd Zhiyun Qian and the anonymous reviewers for their helpful comments. We thank Sebastian Gallenmüller and Eric Osterweil for their support.

REFERENCES

[1] ACM. Result and Artifact Review and Badging. https://www.acm.org/publications/policies/artifact-review-badging, Acc. Jan 18 2017.

[2] 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. Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice. In *22nd ACM Conference on Computer and Communications Security*, 2015.

[3] Devdatta Akhawe, Johanna Amann, Matthias Vallentin, and Robin Sommer. Here's My Cert, So Trust Me, Maybe? Understanding TLS Errors on the Web. In *International World Wide Web Conference*, May 2013.

[4] Alexa. Top 1,000,000 sites. http://s3.amazonaws.com/alexa-static/top-1m.csv.zip, April 02, 2017.

[5] Alexa. Top Sites per Country. http://www.alexa.com/topsites/countries, April 02, 2017.

[6] Johanna Amann. One sided connection tests for Bro. https://github.com/0xxon/bro-onesided-tls, 2017.

[7] Johanna Amann, Robin Sommer, Matthias Vallentin, and Seth Hall. No Attack Necessary: The Surprising Dynamics of SSL Trust Relationships. In *Proc. Annual Computer Security Applications Conference*, 2013.

[8] Johanna Amann, Matthias Vallentin, Seth Hall, and Robin Sommer. Extracting Certificates from Live Traffic: A Near Real-Time SSL Notary Service. Technical Report TR-12-014, International Computer Science Institute, November 2012.

[9] Richard Barnes, Jacob Hoffman-Andrews, and James Kasten. Automatic Certificate Management Environment (ACME). Internet draft, 2017.

[10] Benjamin Beurdouche, Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cédric Fournet, Markulf Kohlweiss, Alfredo Pironti, Pierre-Yves Strub, and Jean Karim Zinzindohoue. A Messy State of the Union: Taming the Composite State Machines of TLS. In *36th IEEE Symposium on Security and Privacy*, May 2015.

[11] Birk Blechschmidt and Quirin Scheitle. GitHub: massdns, commit 4b3148a. https://github.com/quirins/massdns, 2017.

[12] CA/Browser Forum. Ballot 187 – Make CAA Checking Mandatory. https://cabforum.org/2017/03/08/ballot-187-make-caa-checking-mandatory/.

[13] Certificate Transparency. Feb 2014 Survey Responses. Results from Google survey. https://sites.google.com/site/certificatetransparency/feb-2014-survey-responses, 2014.

[14] Chromium. HSTS/HPKP Preload Lists. v389, commit 21f26e9 https://cs.chromium.org/chromium/src/net/http/transport_security_state_static.json?rcl=21f26e9, 02.03.2017.

[15] L. Chuat, P. Szalachowski, A. Perrig, B. Laurie, and E. Messeri. Efficient Gossip Protocols for Verifying the Consistency of Certificate Logs. In *2015 IEEE Conference on Communications and Network Security (CNS)*, 2015.

[16] Taejoong Chung, Yabing Liu, David Choffnes, Dave Levin, Bruce MacDowell Maggs, Alan Mislove, and Christo Wilson. Measuring and Applying Invalid SSL Certificates: The Silent Majority. In *Proc. 16th ACM Internet Measurement Conference (IMC)*. ACM, 2016.

[17] Cisco. Umbrella Top 1M List. http://s3-us-west-1.amazonaws.com/umbrella-static/index.html, April 02, 2017.

[18] Jeremy Clark and Paul C. van Oorshot. SoK: SSL and HTTPS: Revisiting Past Challenges and Evaluating Certificate Trust Model Enhancements. In *Proc. IEEE Security and Privacy*, 2013.

[19] Comodo CA. New certificate for www.fhi.no with corrected SCTs. crt.sh: https://crt.sh/?id=142059145, 2017.

[20] Comodo CA. Revoked certificate for www.fhi.no with invalid SCTs. crt.sh: https://crt.sh/?id=23021029, 2017.

[21] Sergio de los Santos, Carmen Torrano, Yaiza Rubio, and Félix Brezo. Implementation State of HSTS and HPKP in Both Browsers and Servers. In *International Conference on Cryptology and Network Security*. Springer, 2016.

[22] David Dittrich, Erin Kenneally, et al. The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. *US Department of Homeland Security*, 2012.

[23] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J Alex Halderman. Neither Snow Nor Rain Nor MITM...: An Empirical Analysis of Email Delivery Security. In *Proceedings of the 15th ACM SIGCOMM Conference on Internet Measurement Conference (IMC'15)*, October 2015.

[24] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *Proc. 22nd USENIX Security Symposium (USENIX Security)*, 2013.

[25] Chris Evans, Chris Palmer, and Ryan Sleevi. Public Key Pinning Extension for HTTP. RFC 7469 (Proposed Standard), April 2015.

[26] Oliver Gasser et al. GitHub repository for goscanner. https://github.com/tumi8/goscanner, 2017.

[27] Oliver Gasser, Quirin Scheitle, Sebastian Gebhard, and Georg Carle. Scanning the IPv6 Internet: Towards a Comprehensive Hitlist. In *Proc. 8th Int. Workshop Traffic Monitoring and Analysis (TMA)*, 2016.

[28] Google. Certificate Transparency in Chrome. https://a77db9aa-a-7b23c8ea-s-sites.googlegroups.com/a/chromium.org/dev/Home/chromium-security/root-ca-policy/CTPolicyMay2016edition.pdf, May 2016.

[29] Google. Certificate Transparency repository on GitHub. https://github.com/google/certificate-transparency, 2017.

[30] Google. Certificate Transparency website. https://www.certificate-transparency.org/, 2017.

[31] Google and Lexi Brent. GitHub repository for modified version of Google's CT log monitor. https://github.com/lexibrent/certificate-transparency/tree/ctscan_project_dev, 2017.

[32] Josef Gustafsson, Gustaf Overier, Martin Arlitt, and Niklas Carlsson. A First Look at the CT Landscape: Certificate Transparency Logs in Practice. In *Proc. 18th International Conference on Passive and Active Measurement (PAM'17)*, Sydney, Australia, 2017.

[33] Philipp Hallam-Baker and Rob Stradling. DNS Certification Authority Authorization (CAA) Resource Record. RFC 6844, January 2013.

[34] Scott Helme. Getting an A+ on the Qualys SSL Test - Windows Edition. https://scotthelme.co.uk/getting-an-a-on-the-qualys-ssl-test-windows-edition/, Jan 2015.

[35] Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices. In *USENIX Security*, 2012.

[36] Jeff Hodges, Collin Jackson, and Adam Barth. HTTP Strict Transport Security (HSTS). RFC 6797, November 2012.

[37] Paul Hoffman and Jakob Schlyter. The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA. RFC 6698 (Proposed Standard), August 2012. Updated by RFCs 7218, 7671.

[38] Ralph Holz, Johanna Amann, Olivier Mehani, Matthias Wachs, and Mohamed Ali Kaafar. TLS in the wild: An Internet-wide analysis of TLS-based protocols for electronic communication. In S. Capkun, editor, *Network and Distributed System Security Symposium (NDSS'16)*, February 2016.

[39] Ralph Holz, Lothar Braun, Nils Kammenhuber, and Georg Carle. The SSL Landscape: A Thorough Analysis of the X.509 PKI Using Active and Passive Measurements. In *Proc. 11th ACM SIGCOMM Internet Measurement Conference (IMC)*, Berlin, Germany, 2011.

[40] Lin-Shung Huang, Shrikant Adhikarla, Dan Boneh, and Collin Jackson. An Experimental Study of TLS Forward Secrecy Deployments. *IEEE Internet Computing*, 18(6), 2014.

[41] ICANN. Centralized zone data service. https://czds.icann.org, April 02 2017.

[42] Michael Kranch and Joseph Bonneau. Upgrading HTTPS in Mid-Air: An Empirical Study of Strict Transport Security and Key Pinning. In *NDSS*, 2015.

[43] NLnet Labs. Unbound 1.6.0 DNS resolver. https://www.unbound.net, 2017.

[44] Ben Laurie, Adam Langley, and Emilia Kasper. Certificate Transparency. RFC 6962 (Experimental), June 2013.

[45] Eric Law. Internet Explorer should send TLS_FALLBACK_SCSV. https://connect.microsoft.com/IE/feedback/details/1002874/internet-explorer-should-send-tls-fallback-scsv, Oct 2014.

[46] Wilson Lian, Eric Rescorla, Hovav Shacham, and Stefan Savage. Measuring the Practical Impact of DNSSEC Deployment. In *Proc. 22nd USENIX Security Symposium (USENIX Security)*, Washington, D.C., 2013.

[47] Mozilla. HPKP Preload List. https://wiki.mozilla.org/SecurityEngineering/Public_Key_Pinning/Implementation_Details, March 02 2017.

[48] Mozilla. HSTS Preload List. https://wiki.mozilla.org/SecurityEngineering/HTTP_Strict_Transport_Security_(HSTS)_Preload_List, March 02 2017.
[49] Bodo Möller, Thai Duong, and Krzysztof Kotowicz. This POODLE Bites: Exploiting the SSL 3.0 Fallback. https://www.openssl.org/~bodo/ssl-poodle.pdf, 2014.
[50] Bodo Möller and Adam Langley. TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks. RFC 7507 (Proposed Standard), April 2015.
[51] Johnathan Nightingale. Mozilla Security Blog: DigiNotar Removal Follow Up. https://blog.mozilla.org/security/2011/09/02/diginotar-removal-follow-up/.
[52] Notebook http-header-parsing-v4v6, Section Per-Scan-Inconsistency-Analysis. https://mediatum.ub.tum.de/1377982/.
[53] Eric Osterweil, Michael Ryan, Dan Massey, and Lixia Zhang. Quantifying the Operational Status of the DNSSEC Deployment. In *Proc. 8th ACM SIGCOMM Conference on Internet Measurement (IMC)*, 2008.
[54] Craig Partridge and Mark Allman. Ethical Considerations in Network Measurement Papers. *Communications of the ACM*, 2016.
[55] Vern Paxson. Bro: A System for Detecting Network Intruders in Real-Time. *Computer Networks*, 31(23-24), 1999.
[56] PremiumDrops. Domain lists. http://premiumdrops.com/zones.html, April 02 2017.
[57] Ronald Prins. DigiNotar Certificate Authority Breach "Operation Black Tulip". https://www.rijksoverheid.nl/binaries/rijksoverheid/documenten/rapporten/2011/09/05/diginotar-public-report-version-1/rapport-fox-it-operation-black-tulip-v1-0.pdf, September 2012.
[58] Qualy SSL Labs. Ssl server test. https://www.ssllabs.com/ssltest/.
[59] The Register. Blue Coat Chockes on Chrome Encryption. https://www.theregister.co.uk/2017/02/27/blue_coat_chokes_on_chrome_encryption_update/, 27 Februar 2017.
[60] Luigi Rizzo. netmap: A Novel Framework for Fast Packet I/O. In *Proceedings of the USENIX ATC*, 2012.
[61] Luigi Rizzo et al. GitHub repository for lb. https://github.com/luigirizzo/netmap/tree/master/apps/lb, 2017.
[62] Mark D. Ryan. Enhanced Certificate Transparency and End-to-End Encrypted Mail. In *In Network and Distributed System Security Symposium (NDSS). Internet Society*, 2014.
[63] Quirin Scheitle, Oliver Gasser, Patrick Sattler, and Georg Carle. HLOC: Hints-Based Geolocation Leveraging Multiple Measurement Frameworks. In *Network Traffic Measurement and Analysis Conference (TMA)*, Dublin, Ireland, 2017.
[64] Quirin Scheitle, Matthias Wählisch, Oliver Gasser, Thomas C. Schmidt, and Georg Carle. Towards an Ecosystem for Reproducible Research in Computer Networking. In *ACM SIGCOMM 2017 Reproducibility Workshop*, 2017.
[65] Ryan Sleevi. Sustaining Digital Certificate Security. Google blog post: https://googleonlinesecuritys.blogspot.com/2015/12/sustaining-digital-certificate-security.html, 28 October 2015.
[66] Symantec. Certificate Transparency - Symantec Log Server Inclusion. https://bugs.chromium.org/p/chromium/issues/detail?id=483625, 2015.
[67] Symantec. Certificate Transparency for Symantec SSL Certificates. https://knowledge.symantec.com/support/ssl-certificates-support/index?page=content&id=AR2177, 22 August 2016.
[68] Pawel Szalachowski and Adrian Perrig. Short Paper: On Deployment of DNS-based Security Enhancements. *CoRR*, abs/1702.05311, 2017.
[69] Tune The Web. OPINION - Dangerous Web Security Features. https://www.tunetheweb.com/blog/dangerous-web-security-features/, Sep 2015.
[70] Roland van Rijswijk-Deij, Mattijs Jonker, Anna Sperotto, and Aiko Pras. A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements. *IEEE Journal on Selected Areas in Communications*, 2016.
[71] Benjamin VanderSloot, Johanna Amann, Matthew Bernhard, Zakir Durumeric, Michael Bailey, and J Alex Halderman. Towards a complete view of the certificate ecosystem. In *Proc. 16th ACM SIGCOMM Conference on Internet Measurement (IMC'16)*. ACM, 2016.
[72] Verisign. Domain Name Industry Brief Q1'17. https://www.verisign.com/assets/domain-name-report-Q12017.pdf, 2017.
[73] VeriSign Labs. SecSpider—global DNSSEC deployment tracking. http://secspider.verisignlabs.com/stats.html, 2017.
[74] ViewDNS. Domain lists. http://viewdns.info/data/, 2 April 2017.
[75] W3Tech. Usage of web servers for websites. https://w3techs.com/technologies/overview/web_server/all, May 14 2017.
[76] Andrew Whalley and Devon O'Brien. Google Security Blog: https://security.googleblog.com/2017/07/final-removal-of-trust-in-wosign-and.html, 20 July 2017.
[77] Wikipedia. Network Solutions Controversies. https://en.wikipedia.org/w/index.php?title=Network_Solutions&oldid=799250032#Controversies, 26 September 2017.
[78] Kathleen Wilson. Mozilla blog post: https://blog.mozilla.org/security/2016/10/24/distrusting-new-wosign-and-startcom-certificates/, 24 October 2016.
[79] Scott Yilek, Eric Rescorla, Hovav Shacham, Brandon Enright, and Stefan Savage. When Private Keys Are Public: Results from the 2008 Debian OpenSSL Vulnerability. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference (IMC)*, 2009.
[80] Liang Zhang, David Choffnes, Dave Levin, Tudor Dumitras, Alan Mislove, Aaron Schulman, and Christo Wilson. Analysis of SSL Certificate Reissues and Revocations in the Wake of Heartbleed. In *14th ACM Internet Measurement Conference (IMC)*, November 2014.
[81] Liang Zhu, Johanna Amann, and John Heidemann. Measuring the Latency and Pervasiveness of TLS Certificate Revocation. In *Proc. 17th International Conference on Passive and Active Measurement (PAM'16)*, Heraklion, 2016. Springer.
[82] Liang Zhu, Duane Wessels, Allison Mankin, and John Heidemann. Measuring DANE TLSA Deployment. In *Proc. 7th Int. Workshop Traffic Monitoring and Analysis (TMA)*, April 2015.
[83] ZMapv6 on GitHub. https://github.com/tumi8/zmap, May 2017.