A First Look at Certification Authority Authorization (CAA)

Quirin Scheitle[1], Taejoong Chung[2], Jens Hiller[3],
Oliver Gasser[1], Johannes Naab[1], Roland van Rijswijk-Deij[4],
Oliver Hohlfeld[3], Ralph Holz[5], Dave Choffnes[2], Alan Mislove[2], Georg Carle[1]
[1]Technical University of Munich (TUM), [2]Northeastern University, [3]RWTH Aachen,
[4]University of Twente and SURFnet, [5]The University of Sydney
caa@list.net.in.tum.de

ABSTRACT

Shaken by severe compromises, the Web's Public Key Infrastructure has seen the addition of several security mechanisms over recent years. One such mechanism is the Certification Authority Authorization (CAA) DNS record, that gives domain name holders control over which Certificate Authorities (CA) may issue certificates for their domain. First defined in RFC 6844, adoption by the CA/B forum mandates CAs to validate CAA records as of Sep. 8, 2017.

The success of CAA hinges on the behavior of three actors: CAs, domain name holders, and DNS operators. We empirically study their behavior, and observe that CAs exhibit patchy adherence in issuance experiments, domain name holders configure CAA records in encouraging but error-prone ways, and only 4 of the 31 largest DNS operators enable customers to add CAA records. Furthermore, using historic CAA data, we uncover anomalies for already issued certificates.

We disseminated our results in the community. This has already led to specific improvements at several CAs and revocation of mis-issued certificates. Furthermore, in this work, we suggest ways to improve the security impact of CAA. To foster further improvements and to practice reproducible research, we share raw data and analysis tools.

CCS CONCEPTS

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

KEYWORDS

CAA, Web PKI, HTTPS Security

1 INTRODUCTION

Security in the Web critically relies on the SSL/TLS Public Key Infrastructure (PKI) to cryptographically provide authentication and confidentiality. The Web's PKI is rooted in a set of trusted Certificate Authorities (CAs), who issue certificates to domain name holders. A series of mis-issuances [31] shook this fundamental trust and led to various additional security mechanisms: *(i)* Certificate Transparency (CT), a public, append-only log aiming to provide auditable proof of issuance by CAs, *(ii)* HTTP Public Key Pinning (HPKP), now deprecated by Chrome, an HTTP header that allows operators to control the certificates a browser accepts for a website, *(iii)* DANE-TLSA, a DNS-based technology to control certificate use at runtime, and *(iv)* Certification Authority Authorization (CAA), a DNS-based technology to control which CAs may issue certificates for a domain.

CAA is the newest and most recently deployed of these technologies, and we study its early evolution in this paper. In brief, CAA allows a domain name holder to publish a DNS record (called a CAA record) that specifies which CAs—if any—are allowed to issue certificates for that domain. The success of CAA therefore requires the commitment and correct behavior of several stakeholders, all of which we investigate empirically in this study:

Certificate Authorities issue certificates. Per CA/B forum vote, member CAs have committed to respect CAA records as of September 8, 2017 [15]. Using 6 tailored test cases, we examine the issuance process of 12 large CAs in §3.

Domain Name Holders can use CAA records to control which CAs may issue certificates for their domains. Using several longitudinal data sets of large-scale active DNS measurements, we investigate adoption and configuration of CAA records by domain name holders in §4.

DNS Operators are organizations that run authoritative DNS servers. Domain holders can run their own name servers or use external DNS operators, such as the default name servers provided by a registrar. We investigate the extent to which the largest DNS operators—responsible for 54.3% of *.com*, *.net*, and *.org* domains—support CAA records in §5.

Third-Party Auditors can leverage historic CAA records to find anomalies in TLS certificate issuance. This model of third-party scrutiny has been successfully established in Certificate Transparency (CT) and helped to identify various mis-issuances [36, 37]. We take on this role and conduct an end-to-end audit of issued certificates in §6.

Standardization Bodies need to maintain and evolve the CAA standard. These standardization bodies can benefit from our fact-based assessment of the early days of CAA, which we synthesize into specific recommendations in §7.

Taken together, our results present an end-to-end view of how a new security technology is being adopted in its early phase. Despite the relative simplicity of the CAA approach, we find unforeseen challenges and incorrect behavior by various stakeholders. In particular, our key insights are:

- The adherence of CAs to respecting CAA records started with big gaps: For every test we performed, we found at least 1 CA incorrectly issuing a certificate. Re-testing 1 month later showed improvement by multiple CAs.
- The adoption of CAA by domain name holders has steadily grown to over 95k domains. However, we identify non-trivial amounts of misconfigurations and inconsistencies in the published CAA records.

- Over 12% of CAA-enabled domains are DNSSEC signed, compared to ~1% [18] in the general population. This suggests that domains with better security practices are quicker to adopt CAA.
- DNS operators show lackluster behavior in offering CAA to customers whose domains they host. We find ~44% of domain name holders are still unable to set CAA records as their DNS operator does not support it.
- CAA's ability to allow third-party audits allows us to uncover several confirmed mis-issuances (i.e., CAs issuing certificates when CAA records forbid them from doing so); this result proves the value of external evaluation.

Ethical Considerations: In our study, we scrutinized the operational practices of CAs. Similar to prior work [22], we did not seek prior consent, to avoid endangering the validity of our study. We then followed standard procedures in the community by filing public bug reports. This has the advantage that we can name CAs in this paper without risking their reputation. Furthermore, as our test cases were low volume and of a nature previously discussed in the community, it is reasonable to assume that operational stability of CAs was not endangered. These aspects of our study were approved by our respective IRBs. For *active DNS and TLS measurements*, each institution followed established best practices for such measurements as discussed in prior work [2, 23, 25, 38, 48]. We received no complaints about our measurements. We also *notify domain name holders* about anomalous CAA configurations, enabling them to correct their configurations.

Reproducible Research is one of our commitments [2, 42, 43], and we will publish data and analyses upon acceptance.

2 RELATED WORK & BACKGROUND

After a catastrophic breach of DigiNotar in 2011 [40] and further incidents [28, 35, 47], various additional security techniques have been proposed for the Web PKI, which have recently been investigated by Amann *et al.* [2]. For April 2017, they report 216 of the Alexa Top 100k domains to set CAA records, and a total of 3k domains to set CAA records across a large-scale domain scan of 193M domains. One of the data sets used in our paper is a continuation of the measurements by Amann *et al.*. Szalachowski and Perrig [46] find 15 of the Alexa Top 100k domains to set CAA records in August 2016. In a broader context, our work builds on extensive related work on active DNS measurements [2, 27, 41, 48–50], as well as more recent work aimed at understanding the role of domain registrars and DNS operators [19]. For background reading on TLS and Web Security, we recommend [3, 17, 20].

Background: CAA Records provide means for domain name holders to control issuing CAs. Please note that CAA records must not be evaluated by relying parties, *e.g.,* browsers, but are only valid for consumption by CAs at issuance time.

Domain	Type	Flags	Tag	Value
tum.de	CAA	0	issue	"letsencrypt.org"
tum.de	CAA	0	issue	"pki.dfn.de"
tum.de	CAA	0	issuewild	";"
tum.de	CAA	0	iodef	"mailto:a@b"

Table 1: Exemplary CAA section of DNS zone file

Table 1 show an example CAA-enabled zone.CAA records are structured along a *flag*, *tag*, and *value*. Multiple records with the same *tag* form a *set* of *values*. Currently only one flag, the *critical* flag, is defined. This flag instructs CAs not to issue if they do not understand the associated tag. This flag enables future deployment of mandatory tags. The currently defined tags are *issue*, which represents the sets of CAs permitted to issue certificates for a domain, *issuewild*, which optionally overrides the issue set with specific instructions for wildcard certificates, and *iodef*, which defines contact methods for incident information.

In the example from Table 1, only Let's Encrypt and DFN-PKI would be allowed to issue for *tum.de*, and no CA would be allowed to issue a wildcard certificate (*i.e.,* a certificate for `*.tum.de`). Notification e-mails may be sent to *a@b*.

Security Contributions of CAA: On a high level, CAA can help security in three distinct ways: First, CAA can help to actively avoid mis-issuance. We discuss this in detail below. Second, CAA can help to detect mis-issuance through third-party scrutiny (§6). Third, CAA enables domain name holders to bolster their case against mis-issuing CAs by providing zone files or other evidence of their DNS records.

Discussions around RFC 6844 and the CA/B ballots do not define specific attacker models or threat vectors to evaluate the effectiveness of CAA against. CAA records could not have avoided a variety of past mis-issuances [31]. These include CA compromises [40, 53], (sub-)CA malevolence [32, 54, 55], Registrar/TLD compromise [37], or domain side compromise [56]. In these cases, attackers can circumvent checks at the CA or gain control of a domain's DNS infrastructure.

However, in the absence of such a fundamental attack, CAA can improve security posture of a domain by reducing the attack surface: When fewer CAs are authorized to issue for a domain, an attacker has fewer CAs to possibly trick into issuing a fraudulent certificate. CAA records may also help against CA negligence in domain control validation, which has a history of past mis-issuances [26, 28, 35, 47]. This only applies if an otherwise negligent CA properly validates CAA records—which seems achievable, given that exhaustive testing is easier for CAA record validation than for the multitude of domain control validation methods.

Any attacker that can compromise DNS authenticity can easily leverage this to both disable CAA records and validate domain ownership through DNS. Vectors to compromise DNS authenticity are plentiful: Compromise of a TLD, of a domain's DNS infrastructure, or of Internet traffic between validating CA and domain name servers are all effective.

Robustness of CAA against Attacks: While CAA can add a security layer, it is susceptible to transport-based attacks: *Blind Traffic Spoofing:* We consider CAA answers as difficult to spoof in blind off-side attacks. Matching timing, source port, query ID, and query name capitalization requires billions of packets to be sent in a short time, a non-trivial task for an attacker. *Traffic Modification:* If an attacker can modify traffic between a domain's name servers and the querying CA, for example through BGP Hijacking [5], CAA responses can easily be modified or spoofed. *Traffic Corruption:* Even

	Configuration	Expected
D1	signed, restrictive	refuse
D2	signed, timeout	refuse
D3	permissive, but critical unknown	refuse
D4	unsigned, timeout	informational
D5	CNAME to D1	refuse
D6	CNAME to NODATA www.D1	informational

Table 2: Test domains and expected CA behavior.

with capabilities limited to traffic corruption (such as inserting byte errors or flooding links), an attacker can easily disable the use of CAA records, as CAs typically treat lookup failures as permission to issue (§3).

While DNSSEC could protect against these transport-based attacks, CAA in its current version does not mandate DNSSEC checking. The single exception is that lookup failures on signed domains must not be treated as permission to issue—which is frequently not adhered to by CAs (§3).

3 CA SIDE: ISSUANCE EXPERIMENT

To assess whether CAs conform to RFC 6844 [29] and the two CA/B ballots [15, 16], we conduct a set of controlled issuance experiments in two rounds. The first round was conducted in September 2017, when CAA first came into effect. The second round was an extended measurement a month later.

For our experiments, we set up six test domains (D1-D6) that cover various intricacies of the CAA record, such as setting the critical flag, timeouts, and DNSSEC signed zones. For our test domains, we operate two authoritative name servers on which we store all raw traffic.

We use the following definitions in our description: We call a domain *signed* if it has a valid DNSSEC chain to the DNS root. We call CAA records *restrictive* if they do not permit issuance for the CA under test, and *permissive* if they permit issuance. In all cases, we define an `iodef` CAA tag, enabling CAs to report any failed issuance attempt to us—however, we did not receive any such notifications.

We conducted the tests publicly, and informed CAs, the CA/B forum, and Mozilla about our findings and bug reports.

CA Selection: We select Certificate Authorities based on top issuers, as assembled by various sources [24, 49, 51, 52] and the CA/B member list. We prioritize online issuance processes and affordable prices for test certificates. Our final set covers the most significant CAs, issuing 89% of trusted certificates in Censys [24] as of Nov 3, 2017. Choosing the largest CAs probably leads to us erring on the conservative side, as a possible assumption would be that larger CAs have more stable processes. We highlight the complexity of the CA market: Through a variety of brands, sub-CAs, and resellers, the responsible CA is not always easy to decipher. We hence treat any certificate-selling brand as their own CA, even if ownership or infrastructure may be consolidated.

Test Cases: We develop a set of test cases based on RFC 6844, the CA/B ballots [15, 16], and discussion on respective IETF and CA/B mailing lists. Table 2 gives an overview over our test cases: ***D1***, as a signed basic test case, returns a restrictive (`issue ";"`) CAA record, barring any CA from issuing certificates. ***D2*** is a copy of D1, but we

CA ↓ / Expected →	D1 R	D2 R	D3 R	D4 *	D5 R	D6 *
RapidSSL	RR	RI	RR	RI	-R	-I
Comodo	IR	II	IR	II	-R	-I
Let's Encrypt	RR	RR	RR	RR	-R	-I
GoDaddy	RR	RR	RR	II	-R	-I
StartCom	RR	II	RR	RI	-I	-I
Buypass	RR	IR	RR	CI	-R	-R
Certum	RR	IR	RI	II	-I	-I
DigiCert	RR	-R	-R	-I	-R	-I
AlphaSSL	-R	-R	-R	-I	-R	-I
SSL.com	-I	-I	-R	-I	-R	-I
Symantec	-R	-R	-R	-I	-R	-I
GeoTrust	-R	-I	-R	-I	-R	-I

Table 3: Results for CA Issuance Experiments. Per test case and round, we note whether CAs (R)efuse or (I)ssue. Colored backgrounds represent mis-issuances. (C)ancelled denotes cases where a CA cancelled an issuance process upon investigation of other mis-issuances. For example, an entry of RI denotes a CA that in the first round refused to issue, but in the second round mis-issued. A dash (-) denotes that the test case or CA were not included in that round.

silently drop all CAA requests for that domain. CAs must not issue certificates in this timeout case as the zone is signed, which is very specifically highlighted in CA/B Ballot 187 [15]. ***D3*** is unsigned and permissive for each tested CA. However, it returns a record that combines the CAA *critical* flag with an undefined CAA tag. This creates an *unknown critical* record, which denies all CAs from issuing. ***D4*** is unsigned and, as *D2*, drops queries. In this case, CAs may issue certificates if "the lookup has been retried at least once" [15]. This makes this test informational, i.e. any observed behavior is correct. ***D5*** is unsigned, and returns a CNAME pointing to *D1*, which restricts CAs from issuing. ***D6*** is unsigned, and returns a CNAME pointing to the non-existing *www* prefix of *D1*. Erratum 5065 abolishes the need to climb the parent zone at a CNAME target. At the time of our tests, adoption was optional, so this test is informational.

Test Results: Table 3 gives the full overview over our test results. We discuss noteworthy cases in this section:

D1: In our first round, we find Comodo to mis-issue on D1 and all other tests. This was quickly confirmed by Comodo. Root cause analysis revealed that Comodo had a long-standing CAA validation infrastructure, but system updates had silently broken it [11]. In the second round, we could obtain a mis-issued certificate through *SSL.com*. However, *SSL.com* stated that they were a pure reseller of Comodo for this case. This bug is still waiting for root cause analysis [10].
D2 has seen many mis-issuances across both rounds, and was not considered important by some CAs. While this test case may be perceived as a "corner" case, it is one of the specific clarifications the CA/B forum added to RFC 6844 when adopting it through CA/B Ballot 187 [15].
D3 has, besides Comodo not checking CAA at all in the first round, seen only 1 mis-issuance by Certum. Analysis revealed that their implementation depended on record order [8]:

```
CAA 0 issue "certum.pl"
CAA 128 netintum "doesnotexist"
```

The implementation stopped checking further records once

the *issue* tag was seen. As resource records are typically returned in random order, this bug was revealed by chance. ***D5*** has seen mis-issuances from Certum [7] and StartCom [9]. Both confirmed this as mis-issuance.

DNS Lookup Behavior: Using packet captures on our authoritative name servers, we provide in-depth analysis on the CAA query behavior of CAs. To avoid interference from other DNS lookups, such as Internet scans, we conduct a seventh test case, which requests certificates for a random unique query name per CA. RFC 6844 demands that CAs must not rely on DNS data cached by third parties. Furthermore, it states that CAs should deploy "appropriate security controls" to avoid manipulation of CAA records in transport. Our interpretation of appropriate controls could be a distributed lookup infrastructure, querying all authoritative name servers, querying over IPv6 and IPv4, or correlation to third-party lookups. In our experiment, very few CAs deployed any of these security measures: the general pattern observed was one DNS query to one authoritative name server, using IPv4 only. 4 out of 12 tested CAs contact both authoritative name servers. Let's Encrypt and Symantec used variants of query name randomization ("*0x20 DNS*" [21]). Buypass exhibits exemplary request behavior and contacts both our name servers via IPv6 and IPv4, and uses their own resolvers in addition to Google Public DNS. In violation of RFC 6844, one CA relied on (cached) data from OpenDNS. Upon our notification, the problem was quickly acknowledged and fixed.

In conclusion, we note that few CAs deploy security controls on their CAA DNS lookups. This will lead, for example, to inconsistent issuance behavior for the non-trivial number of domains with inconsistent name servers (cf. §4.3).

Discussion: We consider the overall impression from our issuance experiment disheartening for several reasons: First, for every possible test case, we could at least identify one CA trusted by common root stores to mis-issue. An attacker can easily cycle through CAs until finding one that will mis-issue. Second, our bug reports were met by very mixed responses. Some CAs replied quickly and responsibly, and provided incident reports with root cause analysis and mitigation actions. Others dismissed the issues or claimed that they had received DNS responses that permitted their behavior. This claim was upheld without any evidence, despite us providing zone files, packet captures, and stored third party DNS lookups. Escalation through CA/B or Mozilla's NSS bug tracker frequently did not result in timely reaction, either. Third, some CAs actually became worse over time, speaking to a lack of continuous testing. We encourage the CA/B and trust store community to require thorough reports on CAA mis-issuances from CAs.

4 DOMAIN NAME HOLDERS' USE OF CAA

Besides CAs respecting CAA records, adoption by domain name holders is critical to the success of CAA. We conduct large-scale longitudinal scans of large parts of the DNS to evaluate CAA adoption (cf. Table 4). Our scan consists of two phases: First, several disjunct high-volume scans discover domains with CAA records set. Second, the scan TUM-DETAIL ingests all CAA-enabled domains ever discovered in the first phase, and expands them by adding the *www* prefix and extracting all parent domains. The scan is conducted every 8 hours and queries all authoritative name servers per domain. We choose 8 hours as it is the maximum interval that CAs are allowed to cache CAA authorization [15].

Dataset	Labels	Duration (2017)	CAA Domains	Δ
TUM-ZONE [2, 27]	212M	Apr 13 – Nov 08	3k – 41k	1d
OpenINTEL [48]	204M	Oct 28 – Nov 08	37k – 44k	1d
TUM-CT	125M	Sep 07 – Nov 08	14k – 56k	1d
RWTH-ZONE	166M	Sep 08 – Nov 08	14k – 26k	8h
TUM-DETAIL	≈291k	Sep 22 – Nov 08	41k – 95k	8h

Table 4: Datasets and growth of CAA-enabled base domains.

We refer to a zone apex (`tum.de`) directly under a *public suffix* [30, 33](`.de`) as a *base domain*. Any domain ending with a base domain (`www.tum.de`) is referred to as a *label* [30]. As one *base domain* may feature a large amount of *labels*, we usually measure breadth of adoption by *base domains*.

4.1 Growth and Structure

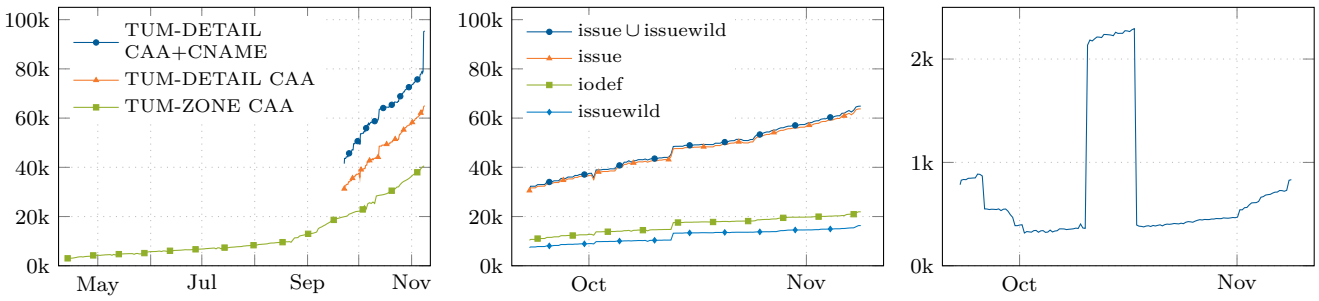
Figure 1a shows a run-up of CAA records for TUM-DETAIL and TUM-ZONE. We can observe three distinct effects: ***(i)***, CAA adoption has been dormant, but has taken up growth since its coming into effect in September 2017. We also note very low churn (not displayed), almost no domain name holders disable CAA. ***(ii)***, the high share of base domains not included in TUM-ZONE implies that a significant amount of domain name holders only configure CAA records on labels below the base domain, using CAA as a fine-grained control mechanism. ***(iii)***, a significant share of base domains is only protected by CAA records when following CNAMEs. This highlights that correct handling of CNAMEs for CAA is of critical importance. CNAMEs are mainly discovered on labels below a base domain, as setting a CNAME at a base domain is generally considered bad practice (cf. RFC 1912, Sec. 2.4). We observe CNAME chain lengths with an average of 1.05 and a maximum of 4, well below the specified limit of 8 [14].

Overall deployment of base domains has reached 65k base domains with CAA records, and 30k base domains with CNAME records leading to CAA records, totaling 95k CAA-enabled base domains as of Nov 8, 2017.

Discontinuities exist in the generally continuous growth. These typically stem from managed hosting companies enabling CAA for their customers' domains. On November 8, managed hosting company *pantheon.io*, enabled CAA on their base domain, which is target of 15k CNAMES.

CAA is broadly deployed: We can report presence within (Top1M: 3k, Top100: 13, Top10: 4) and outside (92k) the Alexa Top1M domains. This speaks to CAA's basic soundness and ease of deployment.

Structural Clustering: To understand the structure of domains that have CAA enabled, we further analyze the domains' *Start of Authority* (SOA) records. SOA records feature a so-called RNAME, which indicates an e-mail address of the responsible zone operator. Clustering by RNAME reveals unexpectedly little centrality, on Nov 08 we see 27k



(a) CAA deployment (b) CAA *tag* deployment (c) NODATA/CAA inconsistency.

Figure 1: Count of base domains over time. Click on Subfigures for extended live versions.

	CA	%	CA	%
1	letsencrypt.org	64.10	";" (no CA)	59.32
2	globalsign.com	7.75	letsencrypt.org	10.57
3	comodoca.com	5.59	thawte.com	5.54
4	symantec.com	4.51	comodoca.com	4.88
5	digicert.com	4.11	globalsign.com	4.15

Table 5: Top 5 values in *issue* (left) or *issuewild* (right) tags.

unique RNAMEs, of which 14% point to Amazon, 4% to Cloudflare and 4% to GoDaddy. In comparison, the total *.com* population sees a 26% share of GoDaddy alone.

This shows that the CAA population is driven by a variety of entities, not just few large hosting companies.

4.2 Deployment Patterns

This subsection investigates which features and configurations of CAA are used by domains, using our comprehensive TUM-DETAIL data set. We find that an encouraging >99% of CAA-enabled base domains use CAA effectively, *i.e.,* set the *issue* or *issuewild* tag. Looking at Figure 1b, we find that 98% of CAA-enabled domains set the *issue* tag, 25% set the *issuewild* tag, and 33% set the *iodef* tag.

Unusual deployments: We find that 2% (1.2k) of domains set unspecified flags, and 12 (0.01%) domains with decode errors, usually due to invalid characters, such as strings in the flags record. 124 (0.1%) domains set unknown CAA tags, typically due to mis-spelling or entering a CA. Fortunately, only 10 of these set the critical flag, restricting any issuance.

Another interesting perspective is the number and type of values (CAs) entered for the *issue* or *issuewild* tags. For the *issue* tag, we typically find a single CA is entered (89%), followed by 2 CAs (6%), 3 CAs (1%) and a long tail of up to 49 CAs. 1.6% of domains allow no CA to issue. This differs strongly from *issuewild*, where 65% do not allow any CA to issue wildcard certificates, 29% allow a single CA, 5% allow two CAs, and the long tail spans to 19 CAs.

When looking at the CAs configured, we surprisingly find CAA used differently than initially expected [39]: Expectations were centered around corporations limiting issuance to commercial CAs with which they have specific agreements, but we find 74% of domains to set Let's Encrypt, a non-commercial CA which is not known to enter customer-specific agreements, in the issue record. Table 5 gives an overview of the top 5 *issue* and *issuewild* CAs.

A worrying matter is the total number of CAs: On Nov. 8, we find ~400 *issue* values and ~130 *issuewild* values. More than 50% of these stem from domain name holders mistakenly entering their own domain name, or from mis-spellings: We see 15 distinct mis-spellings of *letsencrypt.org* alone.

4.3 Name Server Consistency

As discussed in §3, most CAs only query 1 authoritative name server per domain. This means that inconsistent name server configurations will cause inconsistent issuance behavior and can undermine CAA's security contribution. There are two concrete cases in which such behavior can occur. First, the authoritative name servers for a domain may be out of sync, *i.e.,* serve different versions of the DNS zone. Second, there may be different implementations, some not supporting the CAA record type, deployed on the authoritative name servers. In our TUM-DETAIL scan, we query all authoritative name servers for each domain, and find a number of domains to show such inconsistencies: For ~1% of domains, one set of name servers returned a CAA record, while another set returned a NODATA (99%), NXDOMAIN, or CNAME response. Upon further investigation of these NODATA inconsistencies, using SOA serial, name server version, and domain name holder contacts, we frequently find the NODATA name server to run a software version not supporting CAA records yet. This category drastically reduces the protection level provided by CAA on these domains, as CAs usually query only one name server (cf. §3) and will rightfully issue on a NODATA reply.

Aside from this first NODATA category, we also find 17 cases where all servers return a CAA record, but the content of the records differs across servers. Furthermore, for 76 domains the authoritative name servers return different CNAMEs, of which not all lead to CAA-enabled targets.

Figure 1c depicts NODATA/CAA inconsistencies over time. As the figure shows, there are misbehaving domains at any point in time. A spike is visible for one week in October, when 1 out of 4 name servers for a hoster started sending NODATA responses to CAA queries for 1.7k domains. Zone serial and name server set for all affected domains remained unchanged before, during, and after the anomaly. We suspect a software issue at the affected name server.

DNS Operator	CAA Support	% Domains
GoDaddy, Amazon, Google, Cloudflare	✓	44.0%
Alibaba, 1&1, Network Solutions, eNom, Bluehost, NameCheap, WIX, HostGator, NameBright, register.com, OVH, 123-reg, WordPress, Xinnet, DreamHost, Yahoo, Rightside, DNSPod	✗	35.0%
Parking Services	–	21.0%

Table 6: CAA configurable at 4 of the top 31 DNS operators.

4.4 DNSSEC Adoption

As DNSSEC checking is currently not mandatory for CAs [15, 16], it provides little security assurance to domain name holders, except that lookup failure on signed zones must not be interpreted as permission to issue [15]. We investigate whether a significant fraction of CAA-enabled domains use DNSSEC, which could call for making DNSSEC validation mandatory. Among CAA-enabled base domains, 12% have a valid DNSSEC trust chain to the DNS root. This by far exceeds DNSSEC deployment in the general population (~1% [18]). We find this number to have decreased from 18% on Sep. 22, which was not caused by domains disabling DNSSEC, but by the 122% growth of unsigned domains exceeding the 39% growth of signed domains. Our interpretation of these numbers is that CAA records were initially set by security aficionados who would also run DNSSEC, and are now reaching a broader population with a lower share of signed domains.

Also, CAA-enabled domains are more likely to deploy DNSSEC in a functional manner: Only 12% of domains that have a DNSKEY fail to provide a valid trust chain to the DNS root (typically due to missing DS records, in which case the zone can be considered unsigned), whereas about 25% of the general population fail to do so [18]. When discounting missing DS records, only 122 domains (1% of domains with DNSKEY) do not provide validly signed CAA records, a problem of smaller size than inconsistent name servers or misspelled CA names. These results seem to suggest that it will pay off to make DNSSEC validation by CAs mandatory.

5 DNS OPERATORS SUPPORT FOR CAA

In this section, we examine if and how popular DNS operators support CAA records. To do so, we extract domains and corresponding name servers (*i.e.,* `NS` records) from *.com*, *.net*, and *.org* zone files captured on December 31st, 2016. We group domains by the base domain of their `NS` records. For example, we group ns01.bluehost.com and ns02.bluehost.com.

We pick the top 31 DNS operators, covering 54.3% of domains: 20 of the 31 are also registrars, where one can purchase a domain. Two are third-party DNS operators, but not registrars: Cloudflare and DNSPod. The remaining 9 are parking services such as SedoParking. For registrars, we purchase a domain from each registrar and check the possibility to set CAA records on their default name servers, contacting support if this seems not possible. For third-party DNS operators, we use their name servers to see if we can deploy CAA records. We do not further study domain parking.

Anomaly Class	#	TP	Unkn.	FP
Mixed Wildcard	10	4	6	0
Comodo Initial	5	2	3	0
Missing Validation	3	1	1	1
Critical Tags	1	0	1	0
NS Inconsistency	2	n/a	n/a	n/a

Table 7: Issuance Anomalies and confirmation status as True Positive, Unknown/Pending, or False Positive, as of Nov 29, 2017.

Table 6 summarizes the results of this experiment. We immediately notice currently low CAA support: only three registrars (GoDaddy, Amazon, and Google) and one third-party DNS operator (Cloudflare) support creation of CAA records. DNS operators could also minimize misconfigurations by providing a CAA generator or validating customer's inputs. We can confirm that Amazon, Google, and GoDaddy conduct basic validation in their web tool, however we find GoDaddy to do this in a wrong way, *i.e.,* not permitting the only defined flag (128) and instead permitting the undefined "1" flag.

We extrapolate that 44% of used domains on the Internet are not able to configure CAA records on their default name servers, a major obstacle to the success of CAA.

6 END-TO-END AUDIT OF ISSUED CERTIFICATES AGAINST CAA RECORDS

In this section, we take on the role as third-party *auditor* of CAA records as specified in RFC 6844[1] and conduct an end-to-end audit of issued certificates. For domains that set CAA records at least once, we obtain TLS certificates from a variety of sources, and estimate the issuance date from the *not valid before* property and embedded Signed Certificate Timestamps (SCTs). We construct an authoritative mapping of CAA strings to issuing CAs from CA Certificate Policies (CP) and Certificate Practice Statements (CPS). **Limitations** of both analyses are that *(i)*, CAA records may be changed for a very short time period between consecutive measurements on our end, *(ii)* in a split-horizon view, CAs may be presented different responses, and *(iii)*, approximation of CA lookup time from a certificate is coarse: The *not valid before* timestamp of a certificate is usually rounded to the first minute of a day. Embedded SCTs may be more accurate, but are still rare. These limitations prevent third-party auditors from making definite claims about mis-issuances, hence our focus is on uncovering potential anomalies for investigation.

6.1 Issuance Anomalies

When DNS records at the issuance time of a certificate should not have permitted issuance, we call this an *issuance anomaly.* We have reported all of the following cases [34] and received quick confirmations for some, as summarized in Table 7.

Mixed wildcard certificates are certificates that include wildcard and non-wildcard DNS names. It is a common practice by CAs to automatically include a base domain (`tum.de`) in a wildcard certificate (`*.tum.de`). This is, however, not permitted in a CAA configuration like this:

[1]This role is named *evaluator* in RFC 6844.

```
tum.de 0 CAA 0 issue ";"
tum.de 0 CAA 0 issuewild "someCA"
```

While the wildcard issuance for `*.tum.de` is legitimate for *someCA*, validation rules do not permit issuance for `tum.de`. We find 10 mis-issued certificates by 3 CAs for this case, 4 of which have already been confirmed as mis-issuances.

Comodo Initial Problems: As observed in §3, Comodo did not check CAA within the first days of CAA effectiveness. We uncover 5 new mis-issuances for this period.

Missing Validation: For 3 certificates, we find very basic restrictive CAA configurations similar to test case *D1* in §3, suggesting that no validation was done. In one case, the CA could provide evidence that the record had been changed and validated in between our measurements [6].

Critical Tags forbid a CA from issuing certificates for a domain if they do not support a tag with this flag. We find 1 mis-issued certificate for a critical flag, which may be caused by the record being reported as malformed by some DNS lookup tools due to non-printable characters.

Name Server Inconsistency led to 2 issuance anomalies, for which one set of name servers permitted issuance and another set did not. This is not a mis-issuance per se, as the standards do not require CAs to detect or react to this case. However, it proves our point raised in §3, i.e., inconsistent name server configurations can cause insecure issuance behavior.

6.2 Problematic CAA Configurations

In contrast to the issuance anomalies discussed above, we dedicate this section to domain name holders that configure CAA in ways we consider problematic:

Disabling CAA records: In 30 cases, CAA records were permanently disabled at issuance time. Likely domain name holders were not aware of CAA configurations, and disabled those when facing issuance problems. In our issuance experiments, some CAs advised us to do so. We argue that CAs should rather point domain name holders to instructional resources such as CAA record generators [44].

Restricting Renewal: We find 28 cases where domain name holders restrict themselves from renewing their current certificate. We consider these configurations unintentional, as they are either mis-spellings or list several CAs, but not the currently issuing CA. We made domain name holders aware of these anticipated renewal problems.

6.3 Exemplary CAA Configuration

For 1 domain, our scans showed a CAA configuration that consistently did not permit any CA to issue, yet a certificate was issued during this time. Upon inquiry, the domain name holder confirmed that they had fully automated their certificate issuance process, including automatic reconfiguration of CAA records for a brief time period. We consider this case an especially effective security practice, as it restricts any CA from issuing except for few minutes per year.

This confirms the limitation that an auditor's measurements may miss short-term reconfigurations. We consider this a minor limitation to the auditor role when compared to the confirmed anomalies discussed in §6.1.

6.4 Discussion

Our end-to-end audit confirms the value of the auditor role by uncovering several mis-issuances, affecting multiple CAs and several root cause categories. Furthermore, it enables warning domain name holders about pending renewal problems. We consider the uncovering of new *classes* of mis-issuance and the following confirmation and fix by CAs as especially beneficial. As discussed in §1, we have notified affected parties.

7 IMPROVEMENT RECOMMENDATIONS

Backed by our measurement of current CAA practices for all stakeholders, we offer specific improvement recommendations and comment on ideas circulating in the community:

Requiring *iodef* notification for both failed and successful certificate issuances provides domain name holders with the ability to quickly react to attacks. Reliability can be assured by providing email addresses at several different providers. Emails should include all related DNS replies, specifically including the DNSSEC chain or NSEC proofs. Requiring this notification is supported by the point that we have received *no notifications* throughout the dozens of failed issuances in our experiment—the optional status clearly sees no adoption.

Requiring DNSSEC validation provides very strong assurance for signed domains. At 12%, validly signed domains represent a significant share, of which only 1% (a total of 122 domains) invalidly sign their CAA records. The common sentiment that DNSSEC validation would see too many broken domains is clearly not valid for this population.

Restricting Validation Type to a subset of the currently defined 10 types of domain control validation may help domain name holders to restrict validation to types they can strongly secure. As a negative example, email validation has a past of causing mis-issuances [1, 26, 47]. We suggest a *dcv (Domain Control Validation)* tag to whitelist validation methods:

```
tum.de 0 CAA 128 dcv "dns-cname"
tum.de 0 CAA 128 dcv "domain-contact-postal"
```

We also suggest to break down overly broad methods such as "Email, Fax, SMS, or Postal Mail" [13, Sec. 3].

Defining a Minimum Certificate Type of Domain Validated (DV), Organization Validated (OV) or Extended Validation (EV) permits domain name holders to require the scrutiny of OV or EV certificates:

```
tum.de 0 CAA 128 vlevel "ov"
```

In this example, the validation level tag *vlevel* would specify that for *tum.de*, only OV or EV certificates may be issued.

Removal of DNS operator privilege: We consider the privilege of a CA to issue without respecting CAA records if they operate a domain's DNS infrastructure as an unnecessary and dangerous exception. This exception also impedes formal auditing and informal external scrutiny mechanisms such as the auditor role proven so valuable in §6.

Define strategy on name server inconsistency: We have confirmed that non-trivial amounts of domains run inconsistent name servers (cf. §4), CAs usually only check one name server (cf. §3), and that this affects certificate issuance (cf. §4.3). We argue that CAs should form a strategy how to deal with name

server inconsistencies. One such strategy might be to explicitly state that CAs will query only one name server and that domain name holders must assure name server consistency to achieve security goals. A different, more complex strategy might be to query all name servers, and block issuance if a relevant inconsistency is found.

Require DNS Lookup Security Controls: Given that CAA is susceptible to transport-level attacks (cf. §2), we recommend to lower that risk by deploying lookup security controls as discussed in §3. We specifically suggest to probe from several vantage points and use IPv6 and IPv4 where possible.

Require-CT could be introduced as a CAA tag:

```
tum.de 0 CAA 128 requirect "true"
```

Similar to the *expect-ct* HTTP header [45], this could require CAs to submit any issued certificate to at least 2 independently operated CT logs. This would enable domain name holders to assure discovery of mis-issued certificates.

Building an audit record: During our issuance experiments, CAs rarely provided evidence that their issuances were legitimate. We propose requiring CAs to post all CAA lookup results for an issuance process to an append-only ledger similar to Certificate Transparency.

Opposing the use of CAA as a challenge mechanism: Since version 1.5.2, the CA/B forums Baseline Requirements [12] state that CAA records may also be used as part of the challenge/response (C/R) mechanism for DCV issuance. Further ideas include posting a specific CSR to CAA. This dilutes the scope of the CAA mechanism. The goal of CAA is to restrict issuance *ante facto*, whereas the DCV C/R takes place *during* issuance. Using CAA in C/R may wrongly lead domain name holders to believe that they can safely remove a CAA record after successful issuance.

Watch abuse: We foresee scenarios in which bundled CA/Hosting providers will enable a restrictive CAA set as a default "security feature" for their customers. Combined with a complex change process, this could easily drive their hosting customers to their CA business. This case is difficult to regulate, and we encourage watchfulness.

Maintain Tool Support: Given the non-uniform and non-steady validation accuracy of CAs, we urge maintenance of a jointly understood set of test cases as in [4]. Also, CAA record generators such as [44] can reduce the amount of mis-spelling and misconfiguration at domain name holders.

8 CONCLUSION

CAA was voted into effect on September 8, 2017. We have taken an early look at its adoption, effectiveness, and configuration patterns from various stakeholders' perspectives. For CAs, initial support has been so patchy that an attacker could have succeeded for any of our 6 CAA test cases. For domain name holders, we see encouraging adoption in usually reasonable configurations. However, we notice a non-trivial share of mis-spellings, misconfigurations, and security-relevant name server inconsistencies. Furthermore, adoption by domain name holders is inhibited by DNS operators: Only 4 of the large DNS operators that dominate the Internet's DNS infrastructure enable their customers to configure CAA records. We conducted an end-to-end audit of CAA, and found several issuance anomalies. Many of these have been confirmed within days, and fixes deployed by some CAs. This proves the value of the the CAA auditor role. Backed by our data, we have recommended specific improvements for CAA.

Given that our results paint a mixed picture of CAA's success in its early days, we hope that many of our recommendations will be adopted to strengthen CAA. Only time can tell if CAA will lead to actual security improvements, which we intend to study closely in future work

9 ACKNOWLEDGMENTS

We thank supportive individuals from the CA/B community in investigating our reports and reporting back to us.

REFERENCES

[1] Equifax not conforming to Mozilla CA Certificate Policy. https://bugzilla.mozilla.org/show_bug.cgi?id=477783#c19, 2009.
[2] J. Amann, O. Gasser, Q. Scheitle, L. Brent, G. Carle, and R. Holz. Mission Accomplished? HTTPS Security after DigiNotar. In *IMC'17*.
[3] J. Amann, M. Vallentin, S. Hall, and R. Sommer. Extracting Certificates from Live Traffic: A Near Real-Time SSL Notary Service. In *TR-12-014*, 2012.
[4] Andrew Ayer. CAA Test Suite. https://caatestsuite.com/.
[5] H. Birge-Lee, Y. Sun, A. Edmundson, J. Rexford, and P. Mittal. Using BGP to Acquire Bogus TLS Certificates. *HotPETS'17*.
[6] Bugzilla. AlphaSSL CAA Anomaly. https://bugzilla.mozilla.org/show_bug.cgi?id=1420766, Oct 18, 2017.
[7] Bugzilla. Certum CNAME Flag Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1409766, Oct 18, 2017.
[8] Bugzilla. Certum Critical Flag Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1409764, Oct 18, 2017.
[9] Bugzilla. StartCom CNAME Flag Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1409760, Oct 18, 2017.
[10] Bugzilla. SSL.com/Comodo Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1410834, Oct 23, 2017.
[11] Bugzilla. Comodo: CAA Misissuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1398545, Sep 12, 2017.
[12] CA/Browser Forum. Baseline Requirements v1.5.2.
[13] CA/Browser Forum. Baseline Requirements v1.5.4.
[14] CA/Browser Forum. Ballot 214. https://cabforum.org/2017/09/27/ballot-214-caa-discovery-cname-errata/, Nov 10, 2017.
[15] CA/Browser Forum. Ballot 187. https://cabforum.org/2017/03/08/ballot-187-make-caa-checking-mandatory/, Sep 7, 2017.
[16] CA/Browser Forum. Ballot 195. https://cabforum.org/2017/04/17/ballot-195-caa-fixup/, Sep 7, 2017.
[17] T. Chung, Y. Liu, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson. Measuring and Applying Invalid SSL Certificates: The Silent Majority. In *IMC'16*.
[18] T. Chung, R. van Rijswijk-Deij, B. Chandrasekaran, D. R. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson. A Longitudinal, End-to-End View of the DNSSEC Ecosystem. In *USENIX SEC'17*.
[19] T. Chung, R. van Rijswijk-Deij, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson. Understanding the Role of Registrars in DNSSEC Deployment. In *IMC'17*.
[20] J. Clark and P. C. van Oorshot. SoK: SSL and HTTPS: Revisiting Past Challenges and Evaluating Certificate Trust Model Enhancements. In *Proc. IEEE Security and Privacy*, 2013.
[21] D. Dagon, M. Antonakakis, P. Vixie, T. Jinmei, and W. Lee. Increased DNS Forgery Resistance Through 0x20-bit Encoding: SecURItY viA LeET QueRieS. In *CCS'08*.
[22] J. DeBlasio, S. Savage, G. M. Voelker, and A. C. Snoeren. Tripwire: Inferring Internet Site Compromise. In *IMC'17*.
[23] D. Dittrich, E. Kenneally, et al. The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. *US Department of Homeland Security*, 2012.
[24] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman. A Search Engine Backed by Internet-Wide Scanning. In *CCS'15*.

[25] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *USENIX SEC'13*.

[26] Entrust Blog. What Happened with live.fi. https://www.entrustdatacard.com/blog/2015/march/what-happened-with-livefi, Sep 15, 2017.

[27] O. Gasser, Q. Scheitle, S. Gebhard, and G. Carle. Scanning the IPv6 Internet: Towards a Comprehensive Hitlist. In *TMA'16*.

[28] Google Security Blog. Sustaining Digital Certificate Security . https://security.googleblog.com/2015/10/sustaining-digital-certificate-security.html, 2017.

[29] Hallam-Baker, Stradling. RFC6844 – DNS Certification Authority Authorization (CAA) Resource Record, January 2013.

[30] P. Hoffman, A. Sullivan, and K. Fujiwara. DNS Terminology. RFC 7719 (Informational), Dec. 2015.

[31] Ivan Ristic. TLS and PKI History. https://www.feistyduck.com/ssl-tls-and-pki-history/, 2017.

[32] B. Krebs. Turkish registrar enabled phishers to spoof Google. https://krebsonsecurity.com/2013/01/turkish-registrar-enabled-phishers-to-spoof-google/, 2013.

[33] Mozilla. Public Suffix List: commit 85fa8fb. https://github.com/publicsuffix/list/commit/85fa8fbdf, 2017.

[34] Mozilla Security Policy. CAA Anomalies. https://groups.google.com/d/topic/mozilla.dev.security.policy/QpSVjzrj7T4, 2017.

[35] Mozilla Security Policy. Misissued/Suspicious Symantec Certificates. https://groups.google.com/forum/#!topic/mozilla.dev.security.policy/fyJ3EK2YOP8, 2017.

[36] Mozilla Security Policy. ROCA certificate in CT. https://groups.google.com/forum/#!msg/mozilla.dev.security.policy/4RqKdD0FeF4/s5mV8NiqAAAJ, 2017.

[37] Mozilla Security Policy. .tg certificates. https://groups.google.com/forum/#!topic/mozilla.dev.security.policy/4kj8Jeem0EU, 2017.

[38] C. Partridge and M. Allman. Ethical Considerations in Network Measurement Papers. *Communications of the ACM*, 2016.

[39] Paul Hoffman. IETF 80 SAAG Minutes. https://www.ietf.org/proceedings/80/minutes/saag.txt, Mar 31, 2011.

[40] R. 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, Sept. 2012.

[41] J. Rüth, C. Bormann, and O. Hohlfeld. Large-Scale Scanning of TCP's Initial Window. In *IMC'17*.

[42] Q. Scheitle, O. Gasser, P. Sattler, and G. Carle. HLOC: Hints-Based Geolocation Leveraging Multiple Measurement Frameworks. In *TMA'17*.

[43] Q. Scheitle, M. Wählisch, O. Gasser, T. C. Schmidt, and G. Carle. Towards an Ecosystem for Reproducible Research in Computer Networking. In *ACM SIGCOMM Reproducibility'17*.

[44] SSLMate. CAA Generator. https://sslmate.com/caa/.

[45] E. Stark. draft-ietf-httpbis-expect-ct-02.

[46] P. Szalachowski and A. Perrig. Short Paper: On Deployment of DNS-based Security Enhancements. 2017.

[47] R. van Enst. How I got a valid SSL certificate for my ISP's main domain, xs4all.nl. https://raymii.org/s/blog/How_I_got_a_valid_SSL_certificate_for_my_ISPs_main_website.html, 2017.

[48] R. van Rijswijk-Deij, M. Jonker, A. Sperotto, and A. Pras. A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements. *IEEE JSAC*, 2016.

[49] B. VanderSloot, J. Amann, M. Bernhard, Z. Durumeric, M. Bailey, and J. A. Halderman. Towards a Complete View of the Certificate Ecosystem. In *IMC'16*.

[50] T. Vissers, T. Barron, T. Van Goethem, W. Joosen, and N. Nikiforakis. The Wolf of Name Street: Hijacking Domains Through Their Nameservers. In *CCS'17*.

[51] W3Techs. Historical trends in the usage of SSL certificate authorities for websites. https://w3techs.com/technologies/history_overview/ssl_certificate/all, Sep 14, 2017.

[52] WhichSSL. Top 10 SSL Certificate Providers. https://www.whichssl.com/top-10-ssl-certificate-providers.php, Sep 12, 2017.

[53] K. Wilson. Bug 653543—comodo subca. https://bugzilla.mozilla.org/show_bug.cgi?id=653543, 28 April 2011.

[54] K. Wilson. Revoking Trust in one ANSSI Certificate. https://blog.mozilla.org/security/2013/12/09/revoking-trust-in-one-anssi-certificate/, 9 December 2013.

[55] K. Wilson. Revoking Trust in one CNNIC Intermediate Certificate. https://blog.mozilla.org/security/2015/03/23/revoking-trust-in-one-cnnic-intermediate-certificate/, 23 March 2015.

[56] K. Wilson. alicdn.com Misissuance. https://wiki.mozilla.org/CA:WoSign_Issues, June 2016.