# 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 Certification Authorities (CAs) may issue certificates for their domain. First defined in RFC 6844, adoption by the CA/B forum mandates that CAs validate CAA records as of September 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 six 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. The Web's PKI is rooted in a set of trusted Certification Authorities (CAs), who issue certificates to domain name holders. A series of mis-issuances [39] shook this fundamental trust and led to various additional security mechanisms: ***(i)*** Certificate Transparency (CT), a public, append-only log that aims to provide auditable proof of issuance by CAs, ***(ii)*** HTTP Public Key Pinning (HPKP), a HTTP header that allows operators to control the certificates a browser should accept for a website (now deprecated by Chrome), ***(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 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 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:

**Certification Authorities** issue certificates. Per a CA/B forum vote, member CAs have committed to respect CAA records as of September 8, 2017 [18]. Using six 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 their 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 [49, 50]. 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 one CA incorrectly issuing a certificate. Re-testing one 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 numbers of misconfigurations and inconsistencies in the published CAA records.

- Over 12% of CAA-enabled domains are DNSSEC-signed, compared to ~1% [23] in the general population. This suggests that domains actively securing their DNS are quicker to adopt CAA.
- DNS operators show lackluster behavior in offering CAA to customers whose domains they host. We find ~37.4% of domain name holders 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.

To provide an ongoing look at our measurement results for both the research community and industry, we offer a dashboard that continually monitors CAA adoption under

https://caastudy.github.io

**Ethical Considerations:** In our study, we scrutinized the operational practices of CAs. Similar to prior work [27], 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 implies 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, 28, 30, 53, 70]. 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, 60–62, 74], and we publish all code and data under

https://mediatum.ub.tum.de/1403132

Long-term integrity and availability is provided by the University Library of the Technical University of Munich.

## 2 RELATED WORK & BACKGROUND

After a catastrophic breach of DigiNotar in 2011 [56] and subsequent incidents [35, 48, 69], various additional security techniques have been proposed for the Web's PKI. Amann *et al.* [2] recently investigated how these security mechanisms have been deployed. As of April 2017, they report that only 102 of the Alexa Top 100k domains publish CAA records, and a total of 3k domains publish 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 [68] find 15 of the Alexa Top 100k domains to set CAA records in August 2016. Helme [63] conducts daily scans of the Alexa Top 1M domains for several security properties, including CAA; he reports a total of 3.4k CAA-enabled sites as of December 15, 2017. In a broader context, our work builds on extensive related work on active DNS measurements [2, 32, 70–72, 83], as well as more recent work aimed at understanding the role of domain registrars and DNS operators [24]. For a background on TLS and Web Security, we recommend prior work [3, 22, 25].

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

**Timeline of CAA Introduction:** The first draft for CAA stems back to October 2010 [37]. Acceptance in industry took the better part of a decade, shown in Figure 1.

In 2008 and 2009, Zusman, Nigg, and Seifried highlighted that CAs did usually not check if a certificate for a requested domain had already been issued, even if the domain was of high visibility [52, 64, 84]. This sparked the idea of defining authorized CAs for a domain. The IETF standardization process took from October 2010 to publication of RFC 6844 in January 2013. Adoption in the CA/Browser forum took significant time from initial discussion in January 2013 [13] to passing of pre-ballot 125 [15] in October 2014, which requires CAs only to state whether they process CAA records. Further discussions to actually require processing of CAA records ensued in September 2016, resulting in a ballot that makes CAA validation mandatory as of September 2017.

Table 2 compares standardization times for CAA, CT, and HSTS. With 7 years, we find the full-fledged standardization of CAA through IETF and CA/B Forum to take the longest.

| | CAA | CT | HSTS |
|---|---|---|---|
| Initial Draft | 10/2010 | 09/2012 | 06/2010 |
| RFC | 01/2013 | 06/2013 | 11/2012 |
| Enforced | 09/2017 | 05/2015[1] | (01/2010)[2] |

1: CT for EV: [34] 2: Early browser adoption [21, 44]

Table 2: Adoption Timelines of HTTP Security Extensions

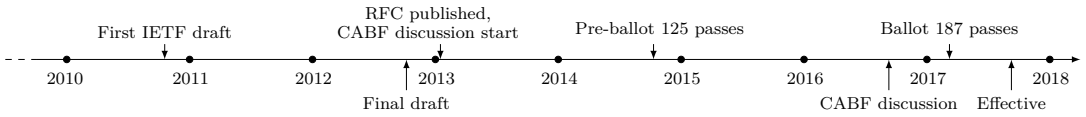



Figure 1: Timeline of CAA standardization, spanning 7 years from initial discussion to introduction as mandatory.

CT and HSTS were standardized in IETF and adopted by browsers without CA/B Forum standardization, permitting for quicker timelines.

**Security Contributions of CAA:** At a high level, CAA can improve security in three distinct ways: First, CAA can help to actively avoid mis-issuance. We discuss this in detail below. Second, CAA can enable third-parties to detect mis-issuance (§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 [39]. These include CA compromises [56, 78], (sub-)CA malevolence [40, 79, 81], registrar/TLD compromise [50], or domain side compromise [80]. In these cases, attackers can circumvent checks at the CA or gain control of a domain's DNS infrastructure. In its current form, CAA also provides little protection against man-in-the-middle attacks between a CA's lookup infrastructure and a domain's name server 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 [31, 35, 48, 69]. 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 who 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 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 [36] and the two CA/B ballots [18, 19], 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 capture and 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 ICANN 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 Certification Authorities based on top issuers, as assembled by various sources [29, 71, 73, 77] 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 [29] as of Nov 3, 2017. Choosing the largest CAs probably leads us to 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 [18, 19], and discussion on respective IETF and CA/B mailing lists. Table 3 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 silently drop all CAA requests for that domain. CAs must not issue certificates in this timeout case as the zone is signed, which is specifically highlighted in CA/B Ballot 187 [18]. ***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,

| | 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 3: Test domains and expected CA behavior.

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" [18]. 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 4 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 [12]. In the second round, we could obtain a mis-issued certificate through *SSL.com*. *SSL.com* stated that they were a reseller of Comodo for this case. Root cause analysis by Comodo revealed that their query had timed out, which they interpreted as permission to issue. Our traffic captures show that our authoritative name servers replied to all queries in the relevant time span [7].

***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 [18].

***D3*** has, besides Comodo not checking CAA at all in the first round, seen mis-issuance only by Certum. Analysis revealed that their implementation depended on record order [10]:

```
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. We will fix the record order to the above for future tests.

***D4*** is an informational test, in which CAs can decide to issue a certificate despite the CAA lookup timeout on an unsigned domain, if "the lookup has been retried at least once" [18]. We can confirm that all CAs have retried DNS lookup at least twice, typically with 20 to 50 retries over several minutes.

***D5*** has seen mis-issuances from Certum [9] and StartCom [11]. Both confirmed this as mis-issuance.

***D6*** shows that CAs, as expected from the problems caused, were quick in abolishing tree climbing for CNAME targets.

| CA ↓ / Expected → | D1 R | D2 R | D3 R | D4 * | D5 R | D6 * |
|---|---|---|---|---|---|---|
| RapidSSL | RR | R[I] | RR | RI | -R | -I |
| Comodo | [I]R | [I][I] | [I]R | II | -R | -I |
| Let's Encrypt | RR | RR | RR | RR | -R | -I |
| GoDaddy | RR | RR | RR | II | -R | -I |
| StartCom | RR | [I][I] | RR | RI | -[I] | -I |
| Buypass | RR | [I]R | RR | CI | -R | -R |
| Certum | RR | [I]R | R[I] | 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 4: Results for CA Issuance Experiments. Per test case and round, we note whether CAs (R)efuse or (I)ssue. Framed red text represents mis-issuances ([I]). (C)ancelled denotes cases where a CA cancelled issuance upon investigation of other mis-issuances. For example, an entry of R[I] 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.

**DNS Lookup Behavior:** Using packet captures from 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, and that CAs should deploy "appropriate security controls" to avoid manipulation of CAA records in transport. Per our interpretation, 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 IPv4 DNS query to one authoritative name server. 4 out of 12 tested CAs contacted both authoritative name servers. However, this might be a performance and not a security choice, and the behavior of CAs under the plethora of possible inconsistencies could be added to test cases in future work.

Let's Encrypt and Symantec used variants of query name randomization ("*0x20 DNS*" [26]). 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, DigiCert 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 from CAs, 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 5) in two phases:

In phase one, several disjunct high-volume scans discover domains with CAA record sets: TUM-ZONE, openINTEL, TUM-CT, and RWTH-ZONE. TUM-ZONE is a high-volume daily scan of about 212M labels (mainly base domains), revealed from zone files and top lists [2, 32]. We report TUM-ZONE separately, as it has a daily coverage of CAA records dating back to April 2017. Furthermore, the set of domains in the scan did not undergo structural changes in the form of addition or removal of zone files. There is still small fluctuation from domain additions and removals in the zone files. The openINTEL [70] and RWTH-ZONE data sets, similar to the TUM-ZONE data set, are both built on various zone files and top lists. The TUM-CT data set consists of about 125M labels found in Certificate Transparency logs.

In phase two, the scan TUM-DETAIL ingests all CAA-enabled domains across all data sets and times 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 [18]. This study includes all data up until November 8, 2017.

We refer to a zone apex (*e.g.,* `tum.de`) directly under a *public suffix* [38, 45](*e.g.,* `.de`) as a *base domain*. Any domain ending with a base domain (*e.g.,* `www.tum.de`) is referred to as a *label* [38]. As a *base domain* may feature many *labels*, we usually measure breadth of adoption in *base domains*.

| Dataset | Labels | Duration (2017) | CAA Domains | Δ |
|---|---|---|---|---|
| TUM-ZONE [2, 32] | 212M | Apr 13 – Nov 08 | 3k – 41k | 1d |
| OpenINTEL [70] | 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 5: Datasets used for this study, growth of CAA-enabled base domains, and measurement interval Δ.

## 4.1 Growth and Structure

Figure 2a 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. For example, on November 8, managed hosting company *pantheon.io* enabled CAA on their base domain, which is target of 15k CNAMEs.

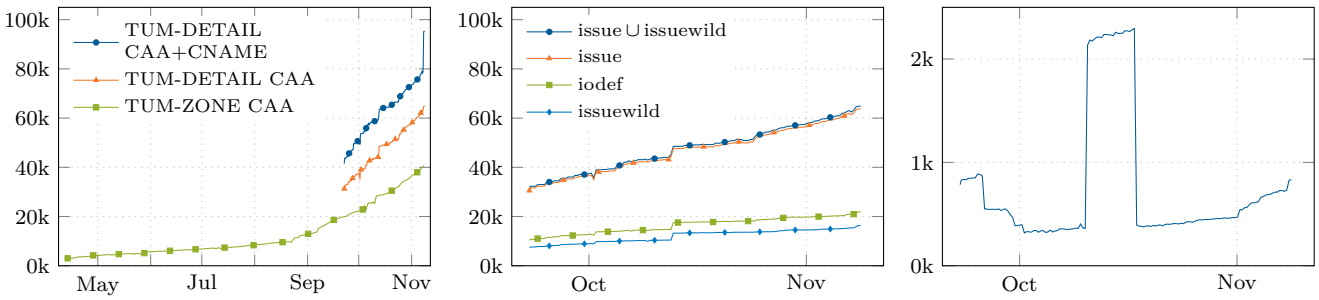
**CAA is broadly deployed**: We can report presence within (Alexa Top 1M: 3k, Top 100: 13, Top 10: 4) and outside (92k) the Alexa Top 1M domains. This speaks to CAA's basic soundness and ease of deployment.

**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: for example, on November 8, 2017 we see 27k 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

We now investigate 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 2b, 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 1.2k (2%) of domains set unspecified flags, which must not be set according to RFC 6844. Records with decode errors occur at 12 (0.01%) domains, usually due to invalid characters, such as strings in the flags field. Unknown CAA tags can be found at 124




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

Figure 2: Count of base domains over time. Click on subfigures for extended live versions

| | *issue* CA | % | *issuewild* 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 6: Top 5 values in *issue* (left) or *issuewild* (right) tags.

(0.1%) domains, typically due to mis-spelling or entering a CA in the *tag* field. Fortunately, only 10 of these set the critical flag and thus restrict 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 [54]: 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 6 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 one authoritative name server per domain. This means that inconsistent name server configurations will cause inconsistent issuance behavior and can undermine CAA's security contributions. 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. For CAs that contact multiple authoritative name servers, it is unclear whether this is for performance or security reasons, and how those CAs would behave under inconsistent replies.

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 2c 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. We note that inconsistent name server configurations have security implications outside the realm of CAA. As the subject of inconsistent name servers is little studied, we also consider this investigation as a pointer to a broader problem.

## 4.4 DNSSEC Adoption

As DNSSEC checking is currently not mandatory for CAs [18, 19], it provides little security assurance to domain name holders, except that lookup failure on signed zones must not be interpreted as permission to issue [18].

With details listed in Table 7, we can report a significant share of CAA-enabled domains to use DNSSEC, exceeding the share in the general domain population by an order of magnitude. Also, we find the share of full DNSSEC deployments (*i.e.,* having a `DS` record in the parent zone) among CAA-enabled domains to be significantly higher than in

| Category | CAA Domains | General Population [23] |
|---|---|---|
| signed (has DNSKEY) | 12% | 1% |
| % without DS record | 12% | 30% |
| % failing validation | 1% | 1% |

Table 7: DNSSEC use among CAA-enabled domains and a general population of domains.

the general population. Signature validation failures among DNSSEC-signed CAA-enabled domains are low and comparable to the general population.

We conclude that a significant share of CAA-enabled domains would benefit from mandatory DNSSEC validation, *i.e.,* requiring a valid `CAA RRSIG` for domains that have a complete ICANN trust chain through `DS` records.

A reason typically brought forward against requiring DNSSEC validation is the allegedly large number of domains with validation errors, which would also affect domains not using CAA records. Our data and the data from [23] suggests that this rate is 1% or less among signed domains, resulting in an estimated 0.001% of domains in the general population with a complete `DS` trust chain that fail DNSSEC validation. For CAA-enabled domains, this group computes to 122 domains, fewer than the 1k domains with inconsistent name servers.

We suggest that CAs refuse certificate issuance with an automated message that DNSSEC validation failed for this small number of domains. Domain name holders can then fix or remove DNSSEC for their domain and re-apply for a certificate. This will improve valid DNSSEC deployment without causing support effort for CAs.

Concluding, we recommend to require correctly signed CAA records (or proofs of non-existence) for domains that have a complete ICANN trust chain through `DS` records. This provides security guarantees to domains that already deploy DNSSEC, it does not make DNSSEC a prerequisite for CAA.

## 5 DNS OPERATOR 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.

Table 8 summarizes the results of this experiment. We immediately notice low support for CAA by registrars: only five registrars (GoDaddy, Amazon, Google, 1&1, and OVH) and one third-party DNS operator (Cloudflare) support creation

| DNS Operator | CAA Support | % Domains |
|---|---|---|
| *T1:* GoDaddy, Amazon, Google, Cloudflare<br>*T2:* 1&1, OVH | ✓ | 49.4% |
| Alibaba, Network Solutions, eNom, Bluehost, NameCheap, WIX, HostGator, NameBright, register.com, 123-reg, WordPress, Xinnet, DreamHost, Yahoo, Rightside, DNSPod | ✗ | 29.6% |
| Parking Services | – | 21.0% |

Table 8: CAA configurable at 6 of the top 31 DNS operators as of February 16, 2018 (T2), up from 4 on November 18, 2017 (T1).

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 incorrectly, by not permitting the only defined *critical* flag ("128") and instead permitting the undefined "1" flag.

We extrapolate that 37.4% of non-parked domains on the Internet are not able to configure CAA records on their current name servers, a major obstacle to the success of CAA. This also reveals the clustering of the global DNS and Web ecosystem around few large entities. Similar to the introduction of other security features, the incentives for these large entities to add CAA support are apparently lacking. Increased DNS operator support could come from increased consumer pull, or unlikely, regulatory push, such as including offering of CAA records into certification criteria for frameworks such as PCI-DSS [55]. A financial incentive from the registry, such as for some TLDs for DNSSEC, is complex, expensive, and unlikely to gain traction: First, experience with incentives for DNSSEC deployment has shown that an incentive for deployment alone does not lead to correct and secure deployment of a technology [42]. Second, checking correct implementation of CAA by DNS Operators is even harder than checking for correct DNSSEC implementation.

## 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 had at least once set CAA records, 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 our 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 than our scans, and *(iii)*, approximation of CA lookup time from a certificate is coarse: The *not valid before* timestamp of a certificate is often

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

| Anomaly Class | # | TP | FP | Unkn. | Pending | Add. |
|---|---|---|---|---|---|---|
| Mixed Wildcard | 10 | 3 | 2 | 4 | 1 | 17 |
| Comodo Initial | 5 | 2 | – | 3 | – | – |
| Missing Validation | 3 | 1 | 1 | 1 | – | – |
| Critical Tags | 1 | – | – | – | 1 | – |
| NS Inconsistency | 2 | n/a | n/a | n/a | n/a | n/a |
| DNS Tree Climbing | – | – | – | – | – | 270 |

Table 9: Issuance Anomalies and confirmation status as True Positive, False Positive, Unknown, or Pending as of March 13, 2018. Additional mis-issued certificates stem from searches for similar mis-issuances conducted by a CA.

rounded in different ways, e.g., to the first minute of a day. Embedded SCTs provide accurate time stamps, but were still rare as of November 2017. We expect them to be deployed in the majority of certificates with the Chrome CT enforcement upcoming in April 2018 [33]. Also, our method cannot reveal whether a domain name holder intended and authorized a CA to issue this certificate, only the domain name holder can assert that. 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 [47] all of the following cases and summarize results in Table 9.

***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:

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

While the wildcard issuance for `*.tum.de` is legitimate for *someCA*, issuance for `tum.de` is not. We find 10 mis-issued certificates by 3 CAs for this case. 2 certificates were false positives: CAs provided logs that showed a permissive CAA record set at their issuance time. The respective domains had apparently changed their CAA record for a brief time period in between our scans. The remaining 7 were issued by Comodo, whose CAA validation was erroneous for this precise case. The error was known internally and a bugfix in preparation. Upon our public inquiry, Comodo started a search for affected certificates. They confirmed this behavior for 17 additional certificates and revoked those. Comodo logged their CAA queries from Oct 12, 2017 onward, so no confirmation of mis-issuance is possible before that date. Of our 7 reported anomalous certificates, 3 were issued past this point and hence revoked, while 4 were issued before that date and hence remain unclassified.

***Comodo Initial Problems:*** As observed in §3, Comodo did not check CAA within the first days of CAA effectiveness. We uncover 5 new anomalous issuances for this period, of which 3 remain unconfirmed due to the lack of logging at Comodo at that point in time.

***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. Two of these were issued by Comodo. One was proven as a false positive due to a brief intermediate record change, and one was issued before Comodo started logging, and hence will remain unconfirmed [6]. The other case revealed a critical error in Camerfirma's interpretation of the CA/B forum's baseline requirements, which led Camerfirma to believe that submitting a precertificate to Certificate Transparency absolved the need for CAA checking. The certificate was revoked, and Camerfirma fixed its CAA validation logic [8].

***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.

***DNS Tree Climbing*** issues were not discovered by us, but we still present them for completeness: Our "missing validation" bug report for Comodo [6] was amended by a domain owner who felt Comodo had mis-issued for his domain. Investigation of this—technically unrelated, but jointly reported—flaw led to discovery of a race condition on Comodo's CAA tree climbing algorithm and revocation of 270 certificates.

## 6.2 Problematic CAA Configurations

In contrast to the anomalies in CA's issuance processes 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 [66].

***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 automated 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.

**Avoiding False Positives:** In our end-to-end audit, we encountered 3 false positives out of 21 reported issuance anomalies. As external auditors are limited to measuring CAA records at scale at most every few hours, short-term reconfigurations of CAA records can cause these effects. This effect was expected, and we diligently investigated each issuance anomaly before reporting. We only reported anomalies fulfilling the following criteria: ***(i)*** we have obtained a stable CAA record before and after the estimated issuance time ***(ii)*** CAA records before and after the estimated issuance time forbid issuance of the certificate ***(iii)*** we were unable to obtain explanation from the domain name holder. In few cases, we decided to forgo criterion (ii) where the facts hinted at a certain known problem (such as mixed wildcard). We encourage future studies to follow the same strict guidelines.

**Nature of Remaining False Positives:** The 3 false positives that occured despite our diligent analysis were rooted in short-term DNS changes in an unexpected manner: Assume that the stable record set observed before issuance is A, the CA-reported issue set at issuance time is B, and the stable record set observed after issuance is C.

In the first case [59], record sets A and C consistently authorized a different than the issuing CA, which reports B to be empty. In the second case [57], record set A did not permit the issuing CA at all, and record set C appeared to intend to authorize the issuing C. Issuance according to issue set C would in fact have been a mixed wildcard mis-issuance. The issuing CA reported a different, authorizing, set B, which apparently was configured for a brief period of time, only to be replaced again with the broken issue set C. The third case [58] also appeared to be a mixed wildcard mis-issuance, but was disproved by the CA with a matching record set B.

**Impact of False Positives:** Due to the limitations discussed above, we reported our findings as issuance anomalies and not as mis-issuances. The 3 resulting false positives were quickly confirmed through CA logs. We consider production of such a log a manual, but reasonably quick, operation. As our end-to-end audit identified several *classes* of issues and led to various fixes at CAs, we consider the overall impact of our end-to-end audit on the CA ecosystem as highly beneficial.

**Sanctioning in the Trust Ecosytem:** Both our issuance experiment and end-to-end audit uncovered serious flaws in CA's processing of CAA records. This raises the question of sanctioning CAs—how to make sure that CAs diligently adhere to market rules?

Trust decisions are taken by individual trust stores such as Mozilla or Google Chrome. These trust stores have their own processes and policies to decide if and to what extent a CA is trusted. The coarseness of trust levels also leads to coarse potential sanctions: Full distrust is a draconic measure that will typically put a CA out of business [65, 76, 82], and a series of full distrusts in a short time period could significantly disturb the HTTPS ecosystem. Few less draconic sanctions are available: ***(i)*** trust store operators can decide to conduct UX sanctions in their browsers. such as the removal of EV indicators. This, however, would further complicate the EV logic and erode benefits it might provide. ***(ii)*** trust store operators can require a CA's certificates to provide CT inclusion promises [65]—this has been done in the past, but becomes ineffective when all certificates will require these promises as of April, 2018. ***(iii)*** trust store operators could establish financial sanctions, but any kind of financial relationship between trust stores and CAs would immediately lead to conflicted interests and mis-aligned incentives.

We argue that due to these unique relationships between CAs, domain name holders, website visitors, trust store operators, and browser vendors, a fine-calibrated quid-pro-quo sanctioning of mis-issuances will be infeasible. We find, however, evaluation of trust to be a diligent process, in which a CA's past mis-issuances are well reflected and discussed. We can direct the reader to the recent re-application of Camerfirma to Mozilla's trust program, which also considers mis-issuances detected by this study [8, 75].

**Relation to CT:** Our end-to-end CAA audit is partially based on CT logs, which can also be used by domain name holders to detect mis-issuances. The expected surge in CT logging in April 2018 [33] will further advance this purpose. Analysis of CT data by third parties can not *assert* issuance against a domain name holder's will; only the domain name holder can assert this. Hence, large-scale analysis of certificate data by third parties typically aims at technical incorrectnesses in CA's issuance processes [41]. Our end-to-end audit aims to uncover technical incorrectness in CAs' issuing processes as well: We correlate certificate data with historic CAA data to detect anomalies in the processing of CAA records by CAs. As CAA data provides information on domain name holders' policies, it enables anomaly detection not possible based on CT's certificate data alone. Consequently, conducting a CAA end-to-end audit at scale can raise issues otherwise undetected by domain name holders or CAs. Ubiquitous CT logging will improve precision and scope of our audit.

## 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 different providers. Emails should include all related DNS replies. Currently, CAs *should* notify iodef contacts in case of rejected issuance. Our experiment, in which we have not received any notification for dozens of failed issuances, proves this current state ineffective.

**Requiring valid signatures for DNSSEC-enabled domains** provides strong assurance for signed domains. Currently, DNSSEC validation is purely optional, and CAs may easily accept forged unsigned responses even for signed domains. At 12%, signed domains represent a significant share of the CAA population. Of these, 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 not valid for this population, and data from [23] suggests it also does not hold true for the general population. Requiring DNSSEC validation for certificate issuance (as CAA checking is now a mandatory part of certificate issuance) will also have a corrective effect on invalid DNSSEC deployments. Please note that we do not suggest making deployment of DNSSEC a prerequisite for CAA: We only want to enforce DNSSEC validation for domains already using DNSSEC.

**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 [31, 43, 69]. We suggest a *dcv (Domain Control Validation)* tag to whitelist validation methods, and a *nodcv* tag to blacklist validation methods:

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

We also suggest breaking down overly broad methods such as "Email, Fax, SMS, or Postal Mail" [16, §3]. January 2018 revealed an interesting case study for such tags: Let's Encrypt was notified of a weakness in the `TLS-SNI` method, and promptly disabled its use. While a following investigation by Mozilla revealed that this method was scarcely used [46] by CAs, there was initial uncertainty about the scope and impact of this weakness. Alert domain name holders may have used our suggested *nodcv* tag to blacklist this method for their domain. Also, as the security weakness is patchable by domain name holders, one suggestion of Let's Encrypt was to let users whitelist this method using CAA. We note that our proposed *dcv* tag may have been used for this.

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

```
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. This could, e.g. be used by domain name holders that exclusively obtain EV certificates, such as financial institutions, to proactively close the attack surface of DV methods. As all security provided by CAA, this only holds true as long as a domain's DNS servers are not compromised.

**Define strategy on name server inconsistency:** We have confirmed that non-trivial amounts of domains run inconsistent name servers (cf. §4), that 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.

**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.

**Require DNS Lookup Security Controls:** Given that CAA is susceptible to transport-level attacks (cf. §2), we recommend lowering that risk by deploying lookup security controls as discussed in §3. We specifically suggest probing 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 [67], 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. With Chrome requiring CT-logged certificates on the client side [33] from April 2018, the value and volume of non-logged certificates will greatly reduce. However, a variety of browsers and TLS clients will not require CT-logged certificates equally soon, so this addition is worth exploring. Please note that, like all CAA tags, this tag is exclusively for consumption by CAs at issuance time. We agree with the community that CAA tags should never be consumed by browsers or other relying parties at connection time.

**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. With the expected logging of all issued certificates to CT, this does not carry additional privacy concerns, and would permit full end-to-end audits without the limitations discussed in §6.

**Opposing the use of CAA as a challenge mechanism:** Since version 1.5.2, the CA/B forums Baseline Requirements [17] 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 express semi-static issuance policies, whereas the DCV C/R takes place for the short time of 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 [66] can reduce the amount of mis-spelling and misconfiguration at domain name holders.

## 8 FUTURE WORK

Reconducting our issuance experiment at a future point may reveal additional longitudinal insight, and we suggest to extend it as follows: First, based on findings throughout our study, we suggest to add test cases for CAA tree climbing and with an intentional ordering of CAA records in query responses. Second, we suggest to extend the scope of CAs tested, specifically by adding smaller CAs. For issuance anomaly detection, we suggest to construct a near-real-time system by coupling a CT export such as Certstream [20] to a DNS scanner and immediately querying all SAN DNS names observed in pre-certificates for their CAA records.

## 9 CONCLUSION

CAA has become effective on September 8, 2017. We have taken a look at its early 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 six 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 six 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, leading to fixes by CAs. 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. We are encouraged by positive feedback from the CA/B forum community on our study and recommendations [51]. Only time can tell if CAA will lead to actual security improvements, which we intend to study closely in future work. We will continuously track CAA adoption and use under https://caastudy.github.io.

## 10 ACKNOWLEDGMENTS

We thank supportive individuals from the CA/B community in investigating our reports and reporting back to us, and SIGCOMM CCR reviewers for their insightful comments. This work was funded in part by NSF grant CNS-1563320, the German Federal Ministry of Education and Research under project X-Check, grant 16KIS0530, and by the DFG as part of the CRC 1053 MAKI.

## REFERENCES

[1] ACM. Result and Artifact Review and Badging. http://acm.org/publications/policies/artifact-review-badging, Jan. 18 2017.
[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/, Sep. 12, 2017.
[5] H. Birge-Lee, Y. Sun, A. Edmundson, J. Rexford, and P. Mittal. Using BGP to Acquire Bogus TLS Certificates. *HotPETS'17*.
[6] Bugzilla. Comodo CAA Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1420873, Jan. 3, 2018.
[7] Bugzilla. SSL.com/Comodo Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1410834, Oct- 23, 2017.
[8] Bugzilla. Camerfirma CAA Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1420871, Oct. 18, 2017.
[9] Bugzilla. Certum CNAME Flag Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1409766, Oct. 18, 2017.
[10] Bugzilla. Certum Critical Flag Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1409764, Oct. 18, 2017.
[11] Bugzilla. StartCom CNAME Flag Mis-Issuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1409760, Oct. 18, 2017.
[12] Bugzilla. Comodo: CAA Misissuance. https://bugzilla.mozilla.org/show_bug.cgi?id=1398545, Sep. 12, 2017.
[13] CA/Browser Forum. CABF Meeting Minutes. https://cabforum.org/pipermail/public/2013-January/001125.html, Jan. 10, 2013.
[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 125. https://cabforum.org/2014/10/14/ballot-125-caa-records/, Oct. 14, 2014.
[16] CA/Browser Forum. Baseline Requirements v1.5.4, Oct. 4, 2017.
[17] CA/Browser Forum. Baseline Requirements v1.5.2, Sep. 20, 2017.
[18] CA/Browser Forum. Ballot 187. https://cabforum.org/2017/03/08/ballot-187-make-caa-checking-mandatory/, Sep. 7, 2017.
[19] CA/Browser Forum. Ballot 195. https://cabforum.org/2017/04/17/ballot-195-caa-fixup/, Sep. 7, 2017.
[20] Cali Dog Security. Certsteam. https://certstream.calidog.io/, Feb. 1, 2018.
[21] Chrome Team. Chrome v4.0.249.78 Release Notes. https://chromereleases.googleblog.com/2010/01/stable-channel-update_25.html, Jan. 25, 2010.
[22] 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*.
[23] 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*.
[24] 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*.
[25] J. Clark and P. C. van Oorshot. SoK: SSL and HTTPS: Revisiting Past Challenges and Evaluating Certificate Trust Model Enhancements. In *IEEE S&P'13*.
[26] 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*.
[27] J. DeBlasio, S. Savage, G. M. Voelker, and A. C. Snoeren. Tripwire: Inferring Internet Site Compromise. In *IMC'17*.
[28] D. Dittrich and E. Kenneally. The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. *US Department of Homeland Security*, 2012.
[29] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman. A Search Engine Backed by Internet-Wide Scanning. In *CCS'15*.
[30] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *USENIX SEC'13*.

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

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

[33] Google. Certificate Transparency Enforcement in Google Chrome. https://groups.google.com/a/chromium.org/d/msg/ct-policy/wHILiYf31DE/iMFmpMEkAQAJ, Feb. 2018.

[34] Google Chrome. Extended Validation in Google Chrome. https://www.certificate-transparency.org/ev-ct-plan, Feb. 07, 2018.

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

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

[37] P. Hallam-Baker, R. Stradling, and B. Laurie. DNS Certification Authority Authorization (CAA) Resource Record. https://datatracker.ietf.org/doc/draft-hallambaker-donotissue/, Oct. 2010.

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

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

[40] B. Krebs. Turkish Registrar Enabled Phishers to Spoof Google. https://krebsonsecurity.com/2013/01/turkish-registrar-enabled-phishers-to-spoof-google/, 2013.

[41] D. Kumar, M. Bailey, Z. Wang, M. Hyder, J. Dickinson, G. Beck, D. Adrian, J. Mason, Z. Durumeric, and J. A. Halderman. Tracking Certificate Misissuance in the Wild. In *IEEE S&P'18*.

[42] T. N. Le, R. van Rijswijk-Deij, L. Allodi, and N. Zannone. Economic Incentives on DNSSEC Deployment: Time to Move from Quantity to Quality. In *NOMS'18*.

[43] G. Markham. Equifax not conforming to Mozilla CA Certificate Policy. https://bugzilla.mozilla.org/show_bug.cgi?id=477783#c19, 2009.

[44] Mozilla. Firefox v4 Release Notes. https://www.certificate-transparency.org/ev-ct-plan, Mar. 22, 2011.

[45] Mozilla. Public Suffix List: commit 85fa8fb. https://github.com/publicsuffix/list/commit/85fa8fbdf, Oct. 28, 2017.

[46] Mozilla NSS. Mozilla January 2018 CA Communication. https://ccadb-public.secure.force.com/mozillacommunications/CACommResponsesOnlyReport?CommunicationId=a051J00003mqMFN&QuestionId=Q00056,Q00057, Feb. 08, 2018.

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

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

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

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

[51] Mozilla Security Policy. Feedback to CAA Study. https://groups.google.com/forum/#!topic/mozilla.dev.security.policy/mqNk9udMwvE, Jan. 10, 2018.

[52] E. Nigg. Unbelievable! https://groups.google.com/d/msg/mozilla.dev.tech.crypto/nAzIKSBEh78/7GEZ4f57F-cJ, Dec. 22, 2008.

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

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

[55] Payment Card Industry . Data Security Standard. https://www.pcisecuritystandards.org, Feb. 20, 2018.

[56] 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, Sep. 5, 2012.

[57] Q. Scheitle. AlphaSSL/Globalsign: CAA Mis-Issuance on mix of wildcard and non-wildcard DNS names in SAN. https://bugzilla.mozilla.org/show_bug.cgi?id=1420766, 2017.

[58] Q. Scheitle. Comodo/cPanel: Potential Mis-Issuance based on CAA records (Sep 28, 2017). https://bugzilla.mozilla.org/show_bug.cgi?id=1420873, 2017.

[59] Q. Scheitle. DigiCert/Thawte: CAA Mis-Issuance on mix of wildcard and non-wildcard DNS names in SAN. https://bugzilla.mozilla.org/show_bug.cgi?id=1420861, 2017.

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

[61] Q. Scheitle, M. Wachs, J. Zirngibl, and G. Carle. Analyzing Locality of Mobile Messaging Traffic using the MATAdOR Framework. In *PAM'16*, Heraklion, Greece.

[62] 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*.

[63] Scott Helme. Tracking CAA Usage. https://scotthelme.co.uk/tracking-caa-usage/, Dec. 15, 2017.

[64] K. Seifried. Breach of trust. http://www.linux-magazine.com/Issues/2010/114/Security-Lessons-Spoofed-Browsers, May, 2010.

[65] R. Sleevi. Sustaining Digital Certificate Security. Google blog post: https://googleonlinesecuritys.blogspot.com/2015/12/sustaining-digital-certificate-security.html, Oct. 28, 2015.

[66] SSLMate. CAA Generator. https://sslmate.com/caa/, Sep. 12, 2017.

[67] E. Stark. Expect-CT Extension for HTTP. https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-expect-ct, Feb. 26, 2018.

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

[69] 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.

[70] 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.

[71] 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*.

[72] 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*.

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

[74] M. Wachs, Q. Scheitle, and G. Carle. Push Away Your Privacy: Precise User Tracking Based on TLS Client Certificate Authentication. In *TMA'17*, Dublin, Ireland.

[75] Wayne Thayer. AC Camerfirma Chambers of Commerce and Global Chambersign 2016 Root Inclusion Request . https://groups.google.com/d/msg/mozilla.dev.security.policy/skev4gp_bY4/snIuP2JLAgAJ, March, 2018.

[76] A. Whalley and D. O'Brien. Google Security Blog: https://security.googleblog.com/2017/07/final-removal-of-trust-in-wosign-and.html, July 20, 2017.

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

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

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

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

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

[82] K. Wilson. Mozilla blog post: https://blog.mozilla.org/security/2016/10/24/distrusting-new-wosign-and-startcom-certificates/, Oct. 24, 2016.

[83] T. Zimmermann, J. Rüth, B. Wolters, and O. Hohlfeld. How HTTP/2 Pushes the Web: An Empirical Study of HTTP/2 Server Push. In *IFIP Networking'17*.

[84] M. Zusman. Domain validated SSL certificates. http://schmoil.blogspot.de/2008/08/domain-validated-ssl-certificates.html, Aug. 25, 2008.

## 11 APPENDIX

We aim to make not only this publication, but our entire study reproducible. Our group is committed to reproducible research [2, 60, 62], which we aim to continue with this study. We structure this section along ***repeatability*** (can the same team obtain the same result when running the same measurement?), ***replicability*** (can an independent team replicate our original results when using our data?) and ***reproducibility*** (can an independent team, using their own tools and measurements, arrive at the same factual conclusion?) [1]. As each aspect requires different prerequisites, we discuss each separately.

### 11.1 Repeatability

Due to the ever-changing nature of the Internet, Internet measurements will yield the exactly same result, hence strict repeatability is challenging. To minimize influence from one-time effects, we leverage our longitudinal data sets with 8-hour measurement frequencies. We argue that the stability of trends and behaviors with little jitter over time in this data set supports the claim that our measurements were repeatable.

### 11.2 Replicability

For others to replicate this paper, we provide all raw data and analyses tools.

We host data and tools at the The University Library of the Technical University of Munich, which assures long-term availability and integrity of our data set under:

https://mediatum.ub.tum.de/1403132

The data set includes documentation on how to replicate our work.

### 11.3 Reproducibility

Reproduction of our results should ideally not be based on our artifacts, but use independent infrastructure. We provide useful detail and high-level guidance for such an endeavor along the sections of this paper:

**Issuance Experiment:** As the issuance behavior of CAs changes over time, a reproduction at a later time may obtain different results. To reproduce our experiment, a team would need two authoritative name servers and a set of 7 domains (we recommend `.com`, some CAs were not able to issue for lesser-known domains). The process consists of generating CSRs for the 7 domains, and then going through the processes of several CAs, trying to obtain certificates for these CSRs. Special attention must be paid to two factors: First, dropping of packets using `iptables` rules must be precisely configured to also match 0x20 (qname randomized) queries, we suggest looking at our proven `iptables` rules to do this. Second, we found that some CAs may refuse to issue based on "improper" CSRs, that might *e.g.,* miss Organizational Unit or Locality values. We recommend creating a verbose CSR and to first test issuance for each CA without restrictive CAA values.

**CAA Records Observed in the DNS:** Our scans are based on zone files, which are available from several Domain Name Registries through individual agreements, and Certificate Transparency, which is publicly available. Most data is measurable using a default scanning tool with a local resolver, such as `massdns` or `zDNS` running against a local `unbound` resolver.

To validate name server consistency, a "detailed" scanner is required that queries all authoritative name servers for a domain. We unfortunately cannot release the proprietary scanner used for this study, but highlight that, for example, `zDNS` can easily be modified to achieve this.

We urge researchers conducting Internet Scanning to follow the best practices laid out in [30].

**Role of DNS Operators:** Reproducing our assessment of how many DNS operators support CAA records requires access to zone files through Verisign[2] to identify top issuers. Further steps include clustering of domains by name servers, identifying the top $n$ DNS operators, obtaining domains from these where possible, and then trying to configure CAA records for these. There is also a list maintained by SSLMate[3].

**End-to-End Audit:** To conduct an end-to-end audit as a CAA auditor requires a thoroughly engineered system with several components. First, an, if possible, historic data set of CAA DNS lookups, possibly several times per day and querying all authoritative name servers. Second, a set of certificates, obtainable, for example, through CT, Censys, or individual scans. Third, a mapping of certificate issuer names to CAA strings. As a starting point, we publish our own mapping, which was engineered from the authoritative CP/CPS documents per CA. Fourth, running code to compare issued certificates to DNS records at a given time.

### 11.4 Website

To further permit our data set to grow into ongoing use, we provide a website with all Figures and Tables from this publication, typically in an extended, interactive, and continuous version. This allows both the exploration of underlying data in more detail, and continuous support to CAA stakeholders:

https://caastudy.github.io

[2] https://www.verisign.com/en_US/channel-resources/domain-registry-products/zone-file/index.xhtml
[3] https://sslmate.com/caa/support