Cryptographically-Secured Domain Validation

Grace H. Cimaszewski*
Princeton University
gcimaszewski@princeton.edu

Henry Birge-Lee*
Princeton University
birgelee@princeton.edu

Cyrill Krähenbühl
Princeton University
cyrill.k@princeton.edu

Liang Wang
Princeton University
lw19@princeton.edu

Aaron Gable
Let's Encrypt
aaron@letsencrypt.org

Prateek Mittal
Princeton University
pmittal@princeton.edu

Abstract

Certificate Authorities (CAs) bootstrap HTTPS-based privacy and trust on the Internet by authenticating the identity of domain names via a process known as domain control validation (DV). Ironically, currently used DV mechanisms rely on unauthenticated web protocols, including plaintext DNS and HTTP, and have been shown vulnerable to a range of network attacks. To address this critical challenge, we propose a framework for cryptographic verification of domain control, which fundamentally mitigates network-layer attacks. Our framework rethinks live DV protocol mechanisms using a domain owner specified security policy which constrains CAs to use authenticated channels and provide cryptographic verification. Our approach minimizes deployment burden on CAs by leveraging existing pieces of the Web PKI and DNS ecosystems, such as Certificate Authority Authorization (CAA) policies and secure DNS. We demonstrate the security properties of our design formally using the Tamarin verification tool and empirically via ethically-conducted real-world attacks. We showcase the feasibility of our framework through collaboration with a major anonymous CA: we analyze DNS and certificate issuance practices of over 400M live domains to understand the current state of CAA policies and secure DNS. We also report on the CA's experiences implementing parts of our design in production. Finally, to realize our framework in the live Web PKI, we led a successful standardization effort for mandatory use of secure DNS by CAs at the CA/Browser Forum.

Keywords

Web PKI, domain control validation, HTTPS privacy

1 Introduction

Obtaining a certificate involves proving control over a domain to a Certificate Authority (CA) through the domain control validation (DV) challenge-response mechanism. This typically involves making specified changes to network resources that presumably only the domain owner can perform, such as publishing a text file containing a nonce on the webserver or a DNS record. However, in practice, DV itself is vulnerable to attacks because it involves unauthenticated protocols, such as DNS and plaintext HTTP as shown in Fig. 1. This core flaw means that certificates' authenticity

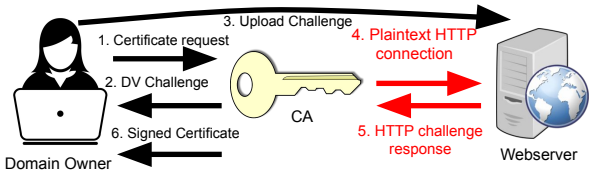


Figure 1: Domain Control Validation via HTTP.

properties can be subverted by network attacks, and attackers can gain certificates for domain names they do not own [9, 12]. Such attacks have already occurred in the wild through BGP or DNS hijacking [8]. This vulnerability has devastating consequences for privacy-enhancing technologies on the Internet, as it can be used to subvert HTTPS protection. An adversary could launch a man-in-the-middle attack and impersonate critical websites, allowing them to intercept private data and tamper with the integrity of tools like VPNs, DoH servers, or private messengers. We are tasked with an important challenge: can we fundamentally mitigate the reliance on untrusted channels to bootstrap trust in today's Web PKI?

The Web PKI security community has responded with a range of solutions which have been adopted to varying extents [10, 32, 44, 54]. *However, none have fully resolved the fundamental challenge of how benign CAs can prevent attacks on their domain validation through cryptographic guarantees against global network adversaries.* Multi-perspective issuance corroboration (MPIC), which queries domain validation challenges from multiple vantage points scattered across the Internet, has been newly adopted by the CA/Browser Forum, the governing body for CAs [15] but provides only heuristic security improvements [10, 14, 19]. Certificate Transparency (CT) provides detection of misissued certificates but does not prevent their issuance [54]. DANE and HPKP give domain owners the ability to more tightly restrict their domains' certificates, but suffer from serious availability problems and never saw long-lived adoption [32, 44]. Finally, several proposals for enhancing PKI security, e.g., by leveraging certificates from multiple CAs, still require that CAs can securely perform domain validation [6, 17, 50, 68, 71, 77].

Innovating in this space has proved particularly challenging because it requires buy-in and careful coordination from three distinct ecosystems: the CAs themselves, the domain owners, and the browser vendors and associated instances that run on many millions to billions of consumer devices and ultimately trust (or reject) certificates. An ideal design provides full security benefits to any domain that chooses to use it; requires minimal to no changes on connecting end-hosts; can be rolled out by CAs in a gradual,

Proceedings on Privacy Enhancing Technologies 2026(2), 452–471

https://doi.org/10.56553/popets-2026-0056

unilateral manner; and does not require extensive modifications to support. At the same time, the design must be downgrade-proof and provide strong security properties across all CAs even when not universally adopted: because all certificates are treated equally by browsers, an attacker can simply choose a "weak link" CA of her choice. The reality of the Web PKI, a collection of roughly 60 distinct commercial CAs, makes the goal even harder. Reasoning about CA behavior is challenging, as technical details of certificate issuance and validation are often CA-specific and not publicly documented. Forcing change across all CAs requires amending the Baseline Requirements issued by the CA/Browser Forum, which lean more towards a description of requirements than a technical specification.

Contributions. In this paper, we propose a framework that meets these qualities while providing cryptographically-secured DV for any opt-in domain. Our design consists of the following components. First, a secure publication and lookup mechanism for *issuance policies*, which are defined by domain owners and verified by CAs. An issuance policy for a domain specifies security constraints that CAs must obey when performing domain validation. Second, we propose a set of security criteria and associated mechanisms for *Cryptographic Domain Validation* (abbreviated as *Cryptographic DV*), that can be implemented by participating (opt-in) CAs. The secure issuance policy paradigm builds upon existing parts of the Web PKI—chiefly, CAA record checking (already performed by all CAs) and authenticated DNS such as DNSSEC validation (already done by many CAs). This approach does not introduce any new infrastructure components, making it not only fully interoperable with current certificate issuance processes but also minimizing the hurdles for its adoption. The secure DNS lookup procedure can also use alternatives such as DoH/DoT, to extend authentication to non-signed DNS records. Cryptographic DV methods then utilize cryptographic keys defined in these policies to mitigate vulnerabilities in current DV mechanisms that rely on insecure channels.

Our work closely builds upon RFC 8657, which introduces DNS CAA extensions for domain owners to specify granular issuance policies to CAs [53]. However, RFC 8657 does not prescribe any framework for cryptographic DV and can be used in insecure ways. We identify several security vulnerabilities in the current usage of existing CAA tags and propose a new CAA `security` tag to mitigate these vulnerabilities (Section 4).

One of the world's largest CAs is implementing parts of our framework in its production deployment, demonstrating our design's practicality (Section 5). We also perform a longitudinal analysis of a dataset from this CA's production deployment logging the issuance of certificates for over 400 million domain names to demonstrate the feasibility of domain owners using our design.

We rigorously analyze the security of our design through a combination of formal and empirical analysis (Section 6). We formally show the security properties of our design using the Tamarin formal verification tool, proving that our design is resilient against any potential message tampering by a Dolev-Yao network attacker. We empirically show the security of our design via ethically-conducted real-world attacks, including BGP attacks as well as adaptive attacks targeting our approach.

Prior to this paper, not all CAs supported authenticated DNS (e.g., DNSSEC validation). Our work has directly led to successful standardization at the CA/Browser Forum to require universal DNSSEC adoption among CAs (Section 5.2). Furthermore, we have written a full specification for the CAA `security` tag, which is under review at the LAMPS working group at the IETF.

We envision a world in which neither rogue hackers nor powerful nation states can target the Web PKI and compromise confidentiality, integrity, and trust in web communications. This paper takes an important step towards this vision by enabling domain owners to fundamentally protect themselves against network-layer attacks on certificate issuance.

2 Background

In this section, we introduce key background technologies that our paper builds on: domain validation, CAA records, the current state of authenticated DNS, as well as lessons learned from failure of prior proposals like DANE and HPKP.

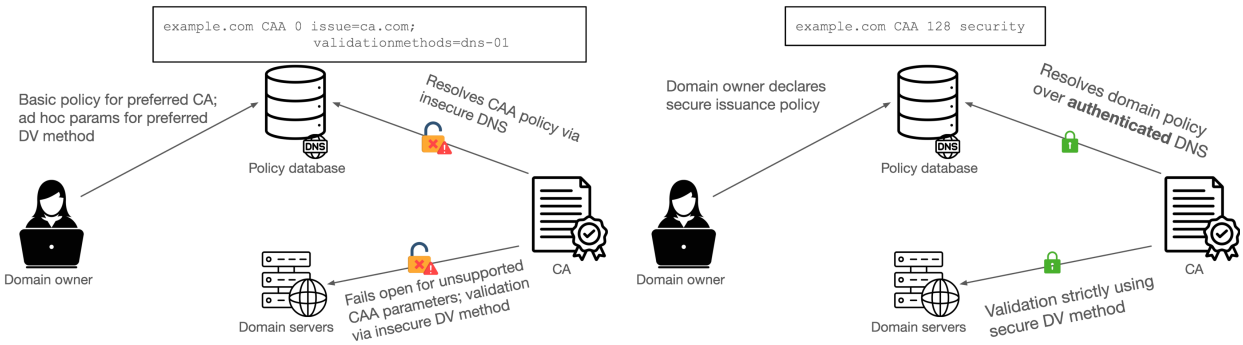
Domain Validation. CAs validate clients' control of domain names through Domain Validation (DV). In this challenge-response protocol, the CA requests the client to place a nonce at a resource (e.g., web server file or DNS record), and then verifies the presence of the requested nonce by querying it. Because querying domain resources involves unauthenticated, plaintext protocols (e.g., DNS or HTTP), DV is vulnerable to network attacks such as MitM attacks. All CAs must perform DV in accordance with the CA/Browser Forum Baseline Requirements for Server Certificates (BRs) to remain trusted by major browsers [16]. Many modern CAs perform DV through the ACME protocol, which standardizes DNS-, HTTP-, and TLS-based DV methods for automated certificate issuance [5].

CAA records. CAA records allow domain owners to specify policies to restrict issuance of their domains' certificates [38]. When validating a certificate signing request, CAs are required to lookup CAA records and ensure that issuance agrees with the specified policy. CAA Records are specified in a `<tag> <value>` format. The most common CAA tag is the `issue` tag, which enumerates the specific CAs that can issue certificates for the domain. RFC 8657 introduced extensions to the `issue` tag which further constrain validation methods that a CA can use. However, RFC 8657 is not standardized at the CA/Browser Forum and its implementation is uneven across CAs.

Authenticated DNS. DNS lookups are vulnerable to tampering attacks that compromise lookup integrity. DNSSEC enables cryptographic authentication of DNS records, allowing DNSSEC-validating resolvers to reject poisoned or tampered domain records [43]. While DNSSEC is a mature standard and supported by major DNS providers, only a small fraction of domains currently use DNSSEC, mainly due to availability concerns and challenges of key rotation [47].

Prior proposals and lessons learned. Several prior proposals aimed to bolster PKI security, but mainly through client-side *post-issuance* integrity checks, not improvements to the DV process.

DNS-based Authentication of Named Entities (DANE) is an IETF-standardized proposal to include PKI information in DNSSEC-signed TLSA records [31, 44]. DANE can operate in several different modes, allowing to specify a set of trusted CAs (mode 0), bypass the traditional PKI entirely by pinning a website certificate (modes 1 and 3), or alternate trust anchor (mode 2). Ultimately, DANE did not



(a) Domain owners configure CAA policies with preferred CAs. Current CAA extensions to specify validation methods may not necessarily correspond to cryptographic guarantees. CAs may use insecure DNS lookups to access CAA policies, subjecting them to attacks. Finally, non-participating CAs may ignore CAA parameters outside of basic "`issue`" directives with no warning.

(b) Domain owners configure CAA policies specifying cryptographic DV (can be agnostic of CA). CAs perform authenticated DNS lookups to access CAA policies, mitigating downgrade attacks. Participating CAs validate domain ownership with cryptographic guarantees. Non-participating CAs fail closed if they do not implement cryptographic DV.

Figure 2: Design comparison between existing workflow for DV and our proposed cryptographic DV in DV failure modes.

see widespread adoption largely due to the challenge of browsers reliably authenticating DNSSEC records [46]. Because DANE records can introduce new trusted certificates for websites, these records *must* be authenticated by DNSSEC, requiring clients to obtain the full DNSSEC chain to verify the records' authenticity. However, nearly 85% of clients use ISP-provided default DNS resolvers, which validate DNSSEC at varying rates and do not relay DNSSEC records to clients, making DANE impractical to use [47].

HTTP Public Key Pinning (HPKP) improves PKI security by implementing a Trust-On-First-Use policy through an HTTP header specifying specific leaf certificates and/or additional certificates (e.g., intermediate certificates) in the certificate chain should be observed for TLS connections to that website [63]. Once this trust on first use is established, clients that have seen the HPKP header will reject any subsequent connections with certificates not matching the pinned one(s), even if it is signed by a trusted authority. HPKP was deprecated due to several fatal flaws: first-time connecting clients that lack pinning information are vulnerable to attacks, and misconfigured HPKP headers or misrotated pinned keys can lead to a denial-of-service attack on the website [40].

3 Design Overview

In this section, we present our approach for cryptographically securing domain validation. It is informed by the deployment shortcomings of prior proposals, DANE and HPKP. The mechanism we propose not only secures domain validation, but is also incrementally-deployable, compliant with CA/Browser Forum requirements, and interoperable with existing systems. Overall, our design involves 1) domain owners uploading a policy in DNS CAA records, 2) DNS operators protecting these policies using `authenticated DNS` (e.g., DNSSEC or recursive-to-authoritative DoT/DoH), 3) CAs performing domain validation with encrypted protocols. The novel components of cryptographic DV are summarized in Fig. 2. It offers security in the presence of non-participating CAs because the CA/Browser Forum requires all CAs to query and respect CAA records.

3.1 Threat Model

Our threat model is a global network adversary which is capable of reading or modifying any communication between clients, servers, and CAs. Recent real-world attacks have involved off-path adversaries that hijack communications (e.g., via BGP attacks) to obtain valid TLS certificates for a domain that they do not control [8, 49]. By considering a global network adversary, our threat model not only considers such off-path BGP attackers (e.g., malicious routers), but also considers on-path attackers that could intercept communications between CAs and servers. This is a strictly stronger threat model than that of MPIC, which focuses only on the narrower threat of equally-specific BGP attacks [10].

Our threat model does not include attacks that allow an adversary to take direct operational control (e.g., via software bugs, misconfigurations, or vulnerabilities) of a victim domain's webserver or its associated DNS server. For example, our threat model does not include attacks on DNS provider credentials, where an adversary maliciously updates a domain's DNS records.

Given the current state of PKI insecurity in which even off-path attackers can obtain TLS certificates from honest CAs, our threat model assumes that CAs are honest, non-compromised, and compliant with all CA/Browser Forum baseline requirements (e.g., those regarding proper CAA checking). We do not consider attacks that involve directly compromising CA infrastructure (e.g., via software/implementation vulnerabilities) to maliciously sign certificates. We discuss how our approach could be augmented to account for malicious CAs in Section F.

To secure DV, our design requires CAs to perform *authenticated* DNS lookups, such as DNSSEC validation. Although authenticating DNS records poses challenges for clients, CAs have the necessary resources to properly perform it. Our threat model assumes that the security properties of DNSSEC or an equivalently secure retrieval

mechanism are not compromised (e.g., via attacks that exploit improper DNSSEC configurations, insecure DNSSEC algorithms, or compromised parent domains that can take over control of the victim's domain). We also assume that DNSSEC-validating resolvers always fail closed (i.e., return SERVFAIL) per RFC specifications [43] if they cannot properly validate a DNSSEC record.

3.2 Design Considerations

By analyzing the pitfalls of existing approaches, we develop several design considerations that inform our approach.

Defense Objective. Our goal is prevent an adversary from obtaining fraudulent certificates via attacks on the domain validation procedures under our threat model. Our defense model aims for attack *prevention*, not merely detection (e.g., Certificate Transparency [2] or constraining certificate usage (e.g., HPKP).)

Design philosophy. As discussed in Section 2, existing techniques like DANE face deployment and interoperability challenges because they require changes to client-side software, such as local DNS resolvers. However, there are billions of clients with heterogeneous systems and software, and ensuring compatibility/security in varied environments is challenging. If even a small fraction of clients fail to opt in properly, it can lead to a major security failure. Therefore, we seek to minimize such changes to facilitate deployment. Our key insights are: (1) CAs possess greater resources and incentives to secure certificate issuance and (2) deployment of a defense at a small number of CAs could have a broad impact on a large number of domains and their clients. Therefore, we transition from a client-centered design to a CA-centered design, by only requiring CAs and domain owners to participate. Our design does not require *any* software changes on client (end-user) devices, including no changes to client-side DNS or TLS software. Such a design pattern provides key benefits in terms of interoperability, incrementally deployability, and key agility.

Interoperability. The design should interoperate with existing technologies and legacy protocols. Clients should be able to engage in secure communication even when using legacy DNS software (a challenge that hindered DANE). Existing webserver software and CDNs should serve websites that opt in to our new protocol without changes to web hosting infrastructure. Finally, our design prioritizes interoperability while mitigating downgrade attacks.

Incremental deployability. The design should be easily deployable and provide security benefits even with partial deployment (e.g., when only a single CA opts in). Additionally, the only global change needed is that CAs perform authenticated DNS lookups. Our work has already achieved standardization for requiring DNSSEC validation (the most established method of authenticated DNS) at the CA/Browser Forum (Section 5.2). With this change in place, our design prevents downgrade attacks against non-participating CAs.

Key agility and availability. A lesson from HPKP's deprecation is that any new improvements to PKI should preserve key agility and availability. Legitimate domain owners should be able to obtain and update certificates even if they lose access to their current credentials such as private keys for existing certificates.

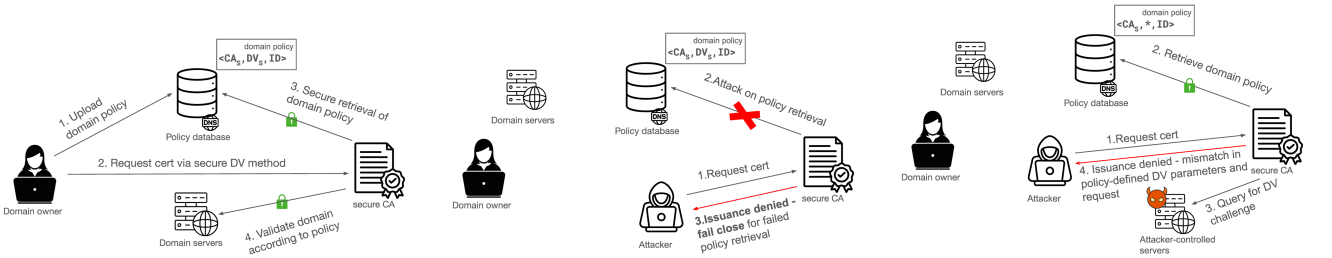
3.3 Cryptographic Domain Validation

To achieve our goal, we develop a framework to embed authenticity and transparency in DV procedures. It enables a CA to verify the integrity of resources relevant to DV to ensure they have not been tampered with (authenticity). It also enables legitimate domain owners to express their DV preferences, such as authorized CAs and preferred secure validation methods, publicly to prevent downgrade attacks that trick a CA into using insecure validation methods (transparency). Furthermore, to minimize deployment burden, our framework bootstraps off existing technologies and is also compatible with CA/Browser Forum requirements. Our contribution lies in shifting away from "adding security" to any existing DV methods, towards an overarching, deployable framework that enables *cryptographically verifying* a users' control over a domain.

Framework components and workflow. Our approach has three major components: (1) domain owners specifying a security policy using DNS CAA records, (2) CAs using an `authenticated DNS` abstraction to securely access domain owner's policy, and (3) CAs performing cryptographic verification of domain control using secure channels. In cryptographic DV (Fig. 3), a participating domain owner specifies a policy that constrains the set of DV methods that a CA is allowed to use to enable cryptographic verification of domain control. When the legitimate user requests a certificate, the CA first securely retrieves the domain policy, and uses the owner-specified policy for DV and certificate issuance. If any attacks within the threat model cause the secure lookup to fail, the certificate issuance is rejected. The CA uses the policy-specified methods to cryptographically validate the user's legitimate control over domain resources using appropriate secure channels. Cryptographic DV harnesses the extensibility of CAA records to build off the existing CAA validation workflow to enable domain owners to specify powerful security properties for their domains' issuances.

How to express and enforce domain owner policies via DNS CAA records, even across non-participating CAs? (§4.1) Certificate Authority Authorization (CAA) records are a type of DNS record [38] that allows domain operators to specify authorized CAs. We use CAA record extensions to enable domain owners to specify security policies. We note that the CA/Browser Forum *mandates* that CAs must perform and adhere to the CAA checking requirement during domain validation ([16], §3.2.2.8). Thus, using CAA records to express secure domain owner policies does not require any changes to the basic certificate issuance workflow, including not requiring any additional communication, additional infrastructure components or external dependencies, or major operational changes by CAs. In Section 4.1, we first analyze opportunities and challenges in using existing CAA record extensions (including RFC 8657) to specify our desired security policy. Motivated by security vulnerabilities that we found in the above approach, we present a new CAA tag extension that robustly specifies a cryptographic security policy and accounts for non-participating CAs.

How to add cryptographic verification to DV mechanisms, while complying with CAB Forum requirements? (§4.2) For bootstrapping cryptographic DV mechanisms, observe that domain owner policies provide a channel of communication between CAs and the domain operator. CAs can thus cryptographically link



(a) To opt into cryptographic DV, the domain owner uploads a secure domain issuance policy to the DNS-based policy database.

(b) An attacker impedes domain policy lookup during a certificate request by blocking or tampering the policy retrieval request/response. Secure lookup fails, and the CA rejects the certificate request.

(c) An attacker tries to bypass the secure DV method(s) specified in the domain's issuance policy. The CA detects a mismatch between the attacker's DV parameters and the policy, and thus blocks issuance.

Figure 3: The secure DV framework (3a) protects against attacks on secure policy lookup (3b), and on DV challenge (3c).

domain control validation to checks performed based on information obtained via this channel. However, such methods must remain compliant with CA/Browser Forum domain control validation practices (Section 3.2.2.4 of the Baseline Requirements for Server Certificate Issuance [16]). We detail how CAs can be CAB-Forum compliant while performing cryptographic verification of domain control Section 4.2. The underlying techniques include (1) CAs cryptographically validating a users' ability to control DNS records (over secure channels) and (2) CAs validating users' control over cryptographic credentials (termed domain *ownerID*), such as public keys or ACME Account URIs) specified in domain owner policies. With cryptographic DV CAA policies in place, a CA fails closed and rejects certificate issuance when it is either non-participating or is unable to cryptographically authenticate the validation.

How to break circular dependencies via `authenticated DNS`? (§4.3). Our proposed components above raise a "chicken-or-egg" dilemma: there is a circular dependency between the PKI's role in bootstrapping secure channels and the reliance of the above components on secure channels for component functionality like domain validation with cryptographic guarantees and securely accessing domain owner policies. To break this circular dependency, our key design insight is that CAs can bootstrap our proposed framework components using already established cryptographic roots as a trust anchor. Notably, we observe that within the existing ecosystem, *authenticated DNS* provides an easily deployable mechanism to serve this purpose. This includes retrieval of DNS records with DNSSEC validation as well as retrieval of DNS records over DoH/-DoT. In Section 4.3, we detail our design choices for `authenticated DNS` and present measurement results from operational logs of an anonymous CA to understand deployability.

4 Using domain owner policies and secure DNS to bootstrap cryptographic DV

In this section, we detail the three main components of our framework: (1) domain owner policies to specify security policies to CAs, (2) cryptographic DV mechanisms that are bootstrapped from these policies, and (3) authenticated DNS techniques that secure both policy lookup and enable cryptographic DV mechanisms. We also present a measurement study of operational logs of an anonymous CA to showcase the prevalence of the building blocks that we rely on, including CAA records and secure DNS.

4.1 Expressing Domain Owner Policies via CAA Records

In current DV, the CA learns the preferred validation method from the certificate request. This allows the adversary to impersonate the domain owner and specify an insecure validation method vulnerable to network attacks, or the adversary could request certificates from a CA that does not support cryptographic DV. To prevent such downgrade attacks, recall that we propose a *domain owner policy* to allow the domain owner to express their preference for cryptographic domain validation. Additionally, the domain owner policy can include *ownerID* cryptographic credentials, such as ACME Account URI, which directly facilitates a CA's cryptographic checks.

CAA record format. We use CAA records to specify the desired security policy. A CAA record [37] consists of (1) Flag: an integer between 0 and 255, which represents two states: non-critical (values $<$ 128) and critical (values $\geq$ 128), 2) Tag: which is a string that defines the action taken by the CA (CAs can define their own tags), and (3) Value: which specifies policies associated with the tag. An example CAA record format with critical flag set to zero and authorizing Let's Encrypt as a CA is: `CAA 0 issue "letsencrypt.org"`.

Leveraging CAA extensibility and critical flag. We build upon two useful features of CAA records. First, the record format is extensible, allowing encoding of more fine-grained policies that enable cryptographic DV. Second, the semantics of the CAA records and CA/Browser Forum requirements mandate that when the critical flag is set, then any CA that does not support the record's tag must refuse issuance (Section 4.5 of RFC 8659 [38]). Thus, the critical flag prevents downgrade attacks against non-participating CAs.

Considerations with CAA validation tree climbing. The CAA record checking is hierarchical and iterates from the specific domain requested in the certificate subject to the parent domains, until a record is found or the root is reached [38]. Thus, to protect a particular subdomain (for example `www.torproject.org`), the closest CAA record associated with this subdomain should specify a secure issuance policy. The closest CAA record could be in the DNS records for the subdomain or parent domain(s). We also note

that a *wildcard* certificate for `*.torproject.org` allows an entity to successfully serve HTTPS connections for *any child subdomain*, including `www.torproject.org`. Thus, in order to rigorously protect subdomains against attacks, our framework requires domain operators to either (1) ensure that certificate issuance for parent domains is also protected by security policies or (2) prohibit wildcard certificate issuance at parent domains. Wildcard certificates can be prohibited using the `issuewild` tag in CAA records: `CAA 0 issuewild ";"` (as done by `torproject.org`).

Next, in Section 4.1.1 we discuss how our policies can be expressed using existing CAA tag extensions but security vulnerabilities arise due to opaque behavior of CAs, non-participating CAs, and misconfigurations. These vulnerabilities motivate our new CAA tag that overcomes these limitations, as detailed in Section 4.1.2.

4.1.1 Opportunities and challenges in existing CAA records. Our proposed domain owner policies can be expressed in RFC-compliant CAA records using the standard `issue/issuewild` tags and overloading desired policy parameters in corresponding `value` field. Although our design can piggyback off an existing DNS record type and component of certificate issuance, practically configuring CAA records to ensure secure issuance for a domain is error-prone due to the complexities of CAA checking and underspecification of CAA security properties. Below, we systematically present our analysis of vulnerabilities in the current usage of CAA records.

CAA processing behavior beyond the basic issue/issuewild tags is overly permissive and varies by CA. An issue tag with the value of a CA name restricts certificate issuance to that specified CA. However, CA/Browser Forum guidelines do not require support of any more sophisticated policies, such as specifying validation methods or *ownerID* credentials. If validation methods and *ownerID* credentials are overloaded onto the issue tag as parameters, their processing is subject to the individual CA's Certificate Practice Statement (CPS), and can change at any time [38]. A CA can silently ignore any other parameters included in the CAA record without any error message or warning to the requester, which does not provide any cryptographic guarantee to domain owners. We experimentally demonstrate this behavior in Section C.

RFC8657 extensions are hard to configure for cryptographically secure issuance, and most CAs don't support them. CAA extensions, described in RFC8657 [53], allow domain owners to overload the `value` field for the standard `issue/issuewild` tags to encode policies for CAs, including ACME account URI (a specific type of *ownerID*) and allowed validation validation methods. Thus, this extension format can be used to express domain owner policies that contain *ownerID* and *validationMethod* corresponding to cryptographic DV. However, RFC8657 provides no framework for specifying secure issuance policies, making it easy for domain owners to misconfigure tags in ways that open up issuance to attack. Of the 33 domains in the Tranco top 10K domains with RFC8657 extensions in their CAA records (protected via DNSSEC), several had policies that still permitted insecure issuance (see Table 6 in Section G). A notable example of a high-stakes domain using RFC8657 is torproject.org. Its CAA records authorize Let's Encrypt with a RFC8657-accounturi parameter, but also authorize issuance by DigiCert and Globalsign without further constraints. This renders the torproject.org domain vulnerable because the unconstrained tags for Digicert and Globalsign allow an attacker to use insecure DV methods and obtain a malicious certificate.

Additionally, RFC8657 extensions are not uniformly implemented by CAs: in fact, most CAs today, including Globalsign, are *non-participating*. RFC8659 CAA checking rules do not define a fail-closed model for partially understood tag values even in the presence of the critical bit. If non-participating CAs are specified in CAA records with *ownerID* and *validationMethod* parameters, CAA checking rules permit these non-participating CAs to ignore the additional parameters and issue even if they do not implement the extensions in the domain's policy (discussed in Section C). This leaves the domain owner's issuance policy unenforced.

Use of insecure CNAME delegation for automated DV. RFC8657 parameters do not formally allow a domain owner to declare their desired *intent* for cryptographic security. Consider a DNSSEC protected domain which specifies a CAA record and uses RFC 8657 to specify the use of a dns-01 [16] based validation method. We uncover a critical vulnerability in this configuration which arises from the use of "ACME magic" CNAME redirection to automate the dns-01 challenge. When a domain owner does not want to integrate automated control of DNS into the webserver initiating the certificate request, they can place a CNAME record under the "`_acme-challenge`" subdomain redirecting to a service that handles the dns-01 challenge on behalf of the client. However, even if the domain being validated is DNSSEC-signed, the target of the CNAME redirection may not be, removing the security guarantees of DNSSEC and exposing the DNS challenge to potential spoofing. We find in our empirical experiments that several cloud providers and DNS management services with automated certificate management offer ACME DNS-style DV delegation, including Fastly [33], AWS Certificate Manager, and Cloudflare [21].

4.1.2 New CAA Tags for Cryptographic DV. We propose a new CAA security tag, that is fundamentally different from the RFC8657 extensions [53] as it enables declarative security in two ways: 1) through use of the critical bit, forcing that non-participating CAs that don't understand the tag or cannot comply *must* not proceed with issuance per CA/Browser Forum rules and 2) specifying truly declarative methods for domain control validation that always use cryptographic primitives and cannot be downgraded.

We require the critical bit to be set in security CAA records. The semantics of the CAA critical bit prevents downgrade attacks in a partial deployment scenario, as non-participating CAs that do not understand the tag must reject issuance. This tag enforces cryptographic DV across all CAs without requiring domain owners to enumerate specific CAs or methods, and is interoperable with existing CAA practices in the CA/Browser Forum Guidelines.

The `security` tag we propose requires that CAs perform cryptographic domain validation when issuing for a domain covered by the tag. The tag optionally allows for domain owners to constrain which methods for cryptographic domain validation should be used. The tag can co-exist with any issue tags and is not degraded if a domain owner includes an issue tag for a CA that does not understand or implement the security tag. Furthermore, the methods supported require cryptographic verification and cannot be downgraded to insecure methods. Table 4 shows that if all CAs perform secure

CAA lookup, e.g., by using `authenticated DNS lookups` to validate the CAA records, setting the critical flag prevents legacy CAs from ignoring the new CAA tag and downgrading the security of the system. We also require the parent domain to also set a security tag-CAA record OR block wildcard issuance to ensure subdomains' CAA records will be checked by the CA during DV.

Use of the `security` tag *enforces* secure issuance, filling the security gaps in standard `issue` tag processing. While existing CAA tags can be used to implement our design, they lack declarative security. Instead, they rely on chaining security properties from various technologies *without allowing domain owners to explicitly declare the desired property of cryptographic DV*. Domain owners must inspect individual CAs' issuance practices and validation methods, determine which of those are secure, and enumerate them in their CAA records, rather than directly communicating the intent of using secure validation methods to CAs. This also places the burden on domain owners to correctly configure their domains' DNS settings and monitor CA issuance practices.

Without using a declarative security tag, domain owners must do ALL of the following to secure their domains' issuance: (1) Manually determine which CAs offer secure issuance by inspecting their CPS for secure issuance practices (e.g., support for RFC8657 extensions), and update DNS records to reflect any changes/churn in the pool of secure preferred CAs. While this limitation is accepted in today's PKI, a resilient future PKI needs automated CA failover, which current practices do not support. The `security` tag causes a hard-fail by non-participating "nonsecure" CAs, thereby protecting domain owners. (2) Ensure that the specified CAA policy satisfies the properties for secure issuance, a non-trivial task in the case of RFC8657 extensions (§4.1.1). The `security` tag comprises only cryptographic DV methods, and domain owners can include it without any value specifiers to secure issuance. (3) Ensure that their domains' DNS supports authenticated lookups, and do not rely on non-DNSSEC-signed nameservers or CNAME delegations to be resolved (§4.3). This is especially challenging in cloud-based environments, where managed TLS certificate services may introduce dependencies on external, non-DNSSEC-signed domain names during domain validation [64]. The `security` tag requires that CAs reject any DV involving unauthenticated DNS lookups, removing the risk of an attacker exploiting insecure DNS.

4.2 CAB Forum-Compliant DV with Cryptographic Verification

Cryptographic information in a domain owner policy can prove authorization to issue a certificate in a manner that is protected against network attackers. For example, a public key can be associated with a domain owner policy and any certificate applicants must show control of the corresponding private key. However, performing CA/Browser Forum compliant domain control validation in a cryptographically-secure way is challenging as some methods for cryptographic proof of ownership (such as the above example) are not compliant with CA/Browser Forum domain control validation practices (Section 3.2.2.4 of the Baseline Requirements for Server Certificate Issuance [16]). To avoid this issue, we leverage the insight that while the baseline requirements specified by the CA/Browser Forum must be followed, including the use of a compliant DV method, a CA is allowed to be more restrictive and perform additional checks on domain ownership. **We use this insight to aid the deployment of cryptographic verification of domain control by proposing methods that can be executed by CAs in addition to CA/Browser-Forum-compliant DV.**

The first set of methods we propose utilize CAs checking for resources at an applicant domain over secure channels. Our proposed secure-dns-record-change consists of performing an ACME "dns-01" challenge or a CA/Browser Forum method 3.2.2.4.7 "DNS Change" *but requiring the relevant DNS records be DNSSEC signed*. This technique offers declarative security whereas these methods generally can be completed over insecure channels. If an existing certificate exists, CAs can use http-validation-over-tls which involves performing an ACME "http-01" challenge or a CA/Browser Forum method 3.2.2.4.18 "Agreed-Upon Change to Website v2" over the HTTPS protocol (via port 443) with verification of the server certificate. http-validation-over-tls is notably not viable for first-time certificate applicants or domains that lost control of their previous key (as discussed by Borgolte *et al.* [11]), but domains in this situation can use one of several other secure methods. With cryptographic DV CAA policies in place, any fallback DV methods must also be cryptographically secure, otherwise a CA would fail closed and rejects certificate issuance.

The second set of methods we propose involve a domain owner uploading cryptographic credentials into their issuance policy. We propose private-key-control where a domain owner puts a public key in a DNS record and authenticates to the CA during the certificate request by showing control of a corresponding private key. This technique has some similarity to DANE [31] but crucially is *checked by the CA* which has more control of its DNS infrastructure and can validate full DNSSEC trust chains. Known-account-specifier, a technique standardized in RFC8657 [53] where a CA-specific account identifier is put into DNS, is a particular instantiation of our general approach. The CA then only accepts certificate requests for that domain from the corresponding account. An advantage of private-key-control over known-account-specifier is that a domain owner can use private-key-control to authenticate itself to any CA whereas the account in known-account-specifier is CA-specific.

Some of our proposed methods build on existing prior work for cryptographic identity verification (e.g., http-validation-over-tls [11] and known-account-specifier [53]). The goal of the our work is not to invent completely new methods for domain control validation but to provide semantics of cryptographically-secure methods for domain owner policies encoded in the security tag.

We note that some of the methods may not strictly comply with the baseline requirements of the CA/Browser Forum. In this case, we propose that these methods be used as an additional cryptographic check performed on top of CA/Browser Forum-compliant DV method. In this manner, we are not constrained by the exact specifications of existing validation methods nor do we need to undergo the lengthy CA/Browser Forum voting process required for updating allowed methods. For example, private-key-control involves a static key in DNS and does not constitute a nonce-based DNS *change* as is required in both ACME "dns-01" or the CA/Browser Forum DNS Change validation method. Our approach allows such a method to provide cryptographic verification while

not requiring any changes to the Baseline Requirements. We discuss another example in Section B.

4.3 Using secure DNS as a trust anchor.

Our framework aims to protect the integrity and authenticity of domain owner policies, ensuring that these policies cannot be created or modified by unauthorized parties and that existing policies cannot be hidden from a CA. We propose the general concept of an `authenticated DNS lookup` which protects CAA lookups from network manipulation.

An `authenticated DNS lookup` represents any DNS lookup procedure that is either signed (i.e., DNSSEC) or performed over a secure channel (i.e., DoH/DoT) with the appropriate authoritative server(s). Our approach leverages authenticated DNS in two ways: first, as a trust anchor to convey domain owners' issuance intent in a secure way, enabling verifiable authorization policies that bind certificate issuance to signed records; secondly, as a policy format itself by using CAA records to encode domain owner policies. This model sidesteps the deployment hurdles experienced by DANE by requiring DNS authentication only on the part of CAs during certificate issuance, not all connecting clients – a much smaller, less latency sensitive-volume of traffic.

Properties of `authenticated DNS lookups`. An `authenticated DNS lookup` mechanism must: (1) retrieve all DNS responses via an encrypted, verified channel or with signature verification to prevent tampering by network-level adversaries; (2) avoid fallback to unauthenticated DNS (i.e., failure to authenticate the DNS lookup is treated as a lookup failure). Finally, presuming not all domains support them, participating domains need to signal their use of `authenticated DNS lookups` in a secure way.

DNSSEC for `authenticated DNS lookups`. DNSSEC is one candidate to achieve these properties. We require CAs to validate DNSSEC to either authenticate the retrieved policy (DNSSEC-signed CAA records) or to validate a proof that no policies exist for this domain (DNSSEC-signed NSEC(3) records [35]) in order to proceed with issuance. If DNSSEC validation fails, the CA must fail-close and deny issuance. Details of our efforts to standardize mandatory DNSSEC validation are included in Section 5.2. We note that requiring CAs to perform DNSSEC checks for DNS lookups only implies authenticity of lookups when corresponding DNSSEC-signed records exist. Thus, requiring CAs to perform DNSSEC checks is a necessary prerequisite to the secure-dns-record-change method which requires that DNS records be DNSSEC-signed (and rejects records in unsigned zones). dns-01 does not suffice because unsigned DNS records are still accepted by a DNSSEC-validating resolver, hence rendering the overall DV insecure.

Secure Channels for `authenticated DNS lookup`. Alternatively, CAs may retrieve the policies via other authenticated channels such as DoT, DoH, and DNSCrypt. For instance, Google recently deployed DoT to authoritative nameservers allowing DoT-based policy retrieval. Nameservers that support secure DNS lookups (i.e., DoH/DoT) should avoid cyclic dependence on Web PKI certificates by distributing details of keys through signed SVCB records [65]. See Section A.1 for more details on how to bootstrap secure channels using SVCB records.

Table 1: Mitigations against attacks on the Web PKI.

Attack	Mitigated by
Forged webserver response	Secure DV methods, *ownerID*
Forged CAA records	`Authenticated DNS lookups`
DoS on CAA records	Hard-fail on `authenticated DNS lookup` failure
Target non-participating CA	Hard-fail on CAA critical bit

Tradeoffs between DNSSEC and Secure channels. Both approaches mentioned above fulfill the necessary security properties needed for `authenticated DNS lookups`. Much of the infrastructure required to support DNSSEC is already in place today. For example, several CAs including Let's Encrypt already validate DNSSEC (Table 2 in Section 5.2), and DNSSEC signing is widely supported by DNS providers. However, a primary challenge facing DNSSEC is that its current adoption by domains hovers around 5% (Section 4.5). Using secure channels for authenticated DNS can potentially protect many more domains with participation from major hosting providers. As DNS hosting is concentrated in a handful of major commercial providers, establishing dedicated tunnels to a small number (as few as 7) of DNS providers can authenticate lookups for a large segment of domains that may not otherwise use DNSSEC (as many as 60%+) [19]. Actually deploying such channels in a manner that satisfies the properties of `authenticated DNS lookups` requires some changes discussed in Section A.

4.4 Achieved Properties and Limitations

With only a single participating CA, domain owners can create cryptographic DV records to achieve rigorous issuance security without sacrificing key agility, with only the overhead of signing DNS records and using a new record tag. **Security.** Every event in the certificate's issuance, from domain policy retrieval to domain control challenge validation, is backed by a cryptographically verifiable chain. Combining secure policy retrieval by all CAs with closed failure for non-opt-in CAs provides security for any participating domain owner, regardless of CA adoption rate. `Authenticated DNS lookups` prevent suppression of the policy, while the fail-close requirement (provided by the CAA record critical bit) ensures that any CA must either use a cryptographic DV method or reject issuance. These two requirements protect against a range of network attacks, summarized in Table 1.

Key agility and availability. Our design ensures that legitimate domain owners can obtain a certificate even when they lose their *ownerID* credentials, private keys for existing TLS certificates, or DNSSEC keys. This is because our design does not exhibit any pinning behavior, a critical weakness in designs such as HPKP. If *ownerID* credentials are lost, a domain owner can simply update its CAA record policy with new credentials. If private keys for existing valid TLS certificates are lost, a domain owner can utilize alternative methods (e.g., secure-dns-record-change) and similarly update the CAA record policy to permit alternative validation methods if needed. In the event a domain owner loses access to its DNSSEC key, the domain can update the DS record hosted at its provider.

Interoperability and incremental deployability. We extend existing CAA records to minimize operational and standardization efforts. Secure policy retrieval only requires a CA to perform `authenticated DNS lookups` for existing CAA queries. Furthermore, cryptographic DV works without changes to legacy webservers since it can be done using secure DNS record changes, ownerIDs, or existing HTTPS sites. Our domain policies only require communication between domain owners and CAs, alleviating the burden and eliminating the need for any additional operations on the clients. Finally, by setting the critical bit in our CAA extensions, we ensure that opt-in CAs can offer tangible security benefits to their customers, while preventing any legacy CA from downgrading the overall security guarantees.

Limitations. There are several hurdles that must be overcome to make cryptographic DV available to a wide range of domain owners. First, to benefit from cryptographic DV, domain owners have to create and upload special CAA tags (e.g., our `security` CAA tag or tag based on RFC 8657) to their DNS nameservers, which limits the initial set of users protected by cryptographic DV. Furthermore, our standardization efforts for the `security` CAA tag are still ongoing, and the tag is currently under discussion at the IETF. Second, cryptographic DV relies on the authenticity of retrieved security tags which depends on the feasibility of `authenticated` DNS lookups. Towards this end, either (1) domain owners must ensure that they enable DNSSEC, which currently has limited deployment, or (2) major DNS nameservers should provide secure channel endpoints for CAs to use, an approach which is still being standardized.

4.5 A deployable approach: measuring DNS practices of domain operators

We measure key trends in how DNS is used by domain operators in the Web PKI, which support DNS's viability both as a trust anchor and policy database. Thanks to our research collaboration with an anonymous CA, we analyzed daily logs of domain validation and certificate issuance events distributed across two datacenter locations in the western U.S. The logs contain certificate request information including validation type, logged responses of CAA record and DV challenge retrievals, and request information (e.g., account URI). We analyzed a 90-day window from Feb. 2024 – May 2024, which encompasses *over 649.2M queries for CAA records* performed for *over 400.8M unique domains*. We also perform DNS queries for domains' DNSKEY records to determine DNSSEC status.

Ethical Considerations. The CA is already required to maintain operational logs per browsers' trusted root program requirements. The CA logged all outbound validation requests, including the requested domain names undergoing domain validation, resolved IP addresses for these domain names, ACME Account URIs, and results of CAA lookups and challenges. Information about domain names and (IP address, CAA, and challenge) DNS records can be publicly resolved through CT logs and DNS, respectively. ACME account IDs do not constitute PII in our dataset, as they appeared as isolated pseudonymous identifiers without accompanying personal data such as registrants' email addresses and names in the logs we analyzed. Finally, we present only aggregate statistics from the logs, except for the case of publicly well-known Tranco-ranked domains.

There is significant adoption of the two main protocol prerequisites, CAA and DNSSEC. Although neither is mandatory for certificate requests, we find that a considerable number of domain owners adopt them in practice. CAA record usage can indicate domain owners' interest in defining security policies to control issuance of their domains' certificates. We find that 13.3% of domains (26.9 M) have a relevant CAA record in the DNS zone hierarchy (i.e., at the domain zone itself or at a parent zone), suggesting CAA's growing popularity and potential future interest in more sophisticated policies. While DNSSEC signing rate remains limited at around 5.88% (11.7M domains), potentially many millions more could utilize cryptographic DV by the usage of secure DNS tunnels to their hosting providers. Domains which already use DNSSEC but do not register CAA records are potential candidates for the new record type, as correctly configuring DNSSEC keys and record signing routines is arguably more involved than creating new CAA records. Domains that already register CAA records and sign them via DNSSEC are "ready-to-go" users in that they can adopt our new extension by simply adding additional tags to their CAA records; we found 689.3K such domains (0.34% of total).

Domain owners increasingly use DNS-based validation and tend to use stable account IDs across certificate renewals, suggesting that they may use cryptographic DV methods smoothly. Our proposed framework can add cryptographic assurance to different validation methods. Non-DNS based validation methods such as ACME-specified HTTP-01 [5] and TLS-ALPN-01 [67] challenges can be secured with accountURI authentication (on top of DNSSEC signing of CAA records). DNS methods such as ACME DNS-01 can rely upon DNSSEC alone for authentication. We studied the distribution of validation methods used by the CA's users over a 90-day period. Encouragingly, we find that **DNS-based DV is popularly used**: 30.8% (123.4 M) of domains used DNS domain validation, 68% (272.5 M) used HTTP, and 0.9% (3.6 M) used TLS-ALPN. These 123M+ domains can, therefore, adopt secure DNS-based validation with minimal changes. We also analyze the pattern of ACME account IDs used to request certificate renewals for domains. ACME account credentials consist of a public-private key pair to authenticate client interactions to a CA. For the *ownerID* policy to be useful, ACME account keys must be treated as long-term account credentials and reused across certificate renewals. We find the **vast majority of domain owners re-use account credentials when renewing certificates**. 93.6% of domains requesting a second certificate 60 days or later after a first certificate (a total of 57.6M domains) used the same account credentials in multiple requests, suggesting that ACME account credentials are not treated as single-use ephemeral request keys.

Domain owners are starting to encode fine-grained issuance policies to CAs. Finally, we inspect the contents of CAA records to gauge domain owners' interest in bespoke security-oriented issuance policies. A small fraction of domains already constrain CA behavior by specifying ACME accountURI and/or acceptable validation methods in a DNSSEC-signed CAA record (Fig. 4). 74.8K domains specify a DNS-based DV method or account ID in a DNSSEC-signed CAA record, making them "ready-to-go" domains in our framework (e.g., using our proposed security tag). These domains include Debian Linux, the Slack messaging app, and 92 .gov domains such as max.gov, omb.gov, and cio.gov. A more detailed analysis of

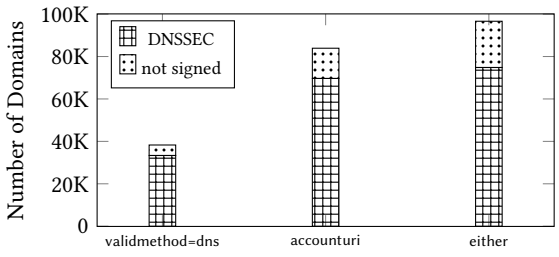


Figure 4: Usage of secure issuance policy elements in CAA.

such high-impact domains is included in Appendix G. Of note is domain owners' preference for using account ID credentials: over 93% (69.7K) of such domains specified account credentials in their issuance policies. Finally, we highlight the high rate of DNSSEC adoption among these early adopters: over 77.4% of the 96.7K domains that constraint CA behavior are DNSSEC-signed, suggesting a heightened awareness of DNS security practices.

In summary, widespread use of DNS-based validation, high CAA record usage among active domains, and stable reuse of account credentials provide empirical support for using these building blocks in our design. They also reinforce DNS's suitability as a trust anchor for secure issuance policies, as it is already queried by CAs, familiar to domain operators, and has options for authenticated transport. Cryptographic DV builds on current practice for a deployment-ready path to cryptographically-secure certificate issuance.

5 Preliminary Deployment and Standardization

In this section, we present the experiences of a major anonymous CA that implements partial features of *cryptographic DV* via the mechanisms in Section 4.1. We also describe our successful standardization work at the CA/Browser Forum and our ongoing standardization efforts at the IETF.

5.1 Deployment Experiences at Anonymous CA.

The CA (1) implements *authenticated DNS lookups*; (2) implements recognition of *validationMethod* and *ownerID* domain owner policies within CAA records, and (3) implements the "known-account-specifier" validation protocol as described below.

Deploying authenticated DNS lookups. The CA validates DNSSEC for all DNS queries, both for CAA retrieval and domain control validation. As a publicly trusted CA, they are required by the CA/Browser Forum to operate their own recursive resolvers. In the CA's 9+ years of using Unbound, DNSSEC validation has not caused any availability or correctness issues. The CA aims to deploy authenticated DNS via secure channels in the near future.

Implementing authenticated domain owner policies. The CA implements support for RFC 8657 [53], which specifies two parameters which domain operators can add to their existing CAA records. To the best of our knowledge, this is the first production deployment of ACME CAA. The "accounturi" parameter allows the CAA record to specify a single ACME account which is authorized to request issuance for the domain. This corresponds to specifying an *ownerID* policy. Similarly, the "validationMethods" parameter allows the CAA record to specify one or more ACME validation methods which can be used to validate control of the domain, and corresponds to specifying a *validationMethod* policy.

Implementation of support for these CAA parameters took about two engineer-weeks, and consumes about 100 lines of Go code. One intricacy of note is that, because the CA's CAA checking routines simply return a "issuance (dis)allowed" value, all relevant values (the unique identifier of the account requesting issuance, the name of the validation method actually used, etc.) must be passed *in* to the CAA checking routine so that they can be compared against the values found in the site's CAA records. Other implementers may prefer to return a data structure of the policy described by the CAA records, and compare that policy against the requesting account and validation method elsewhere in the issuance system.

Implementing cryptographic DV. By combining the two elements above, the CA enables domain operators to use a limited form of cryptographic DV. To opt in, the domain operator (1) enables DNSSEC, (2) sets a CAA record with the accounturi parameter specifying their unique account identifier, (3) does not CNAME delegate the ACME challenge subdomain, or ensures that any CNAME chain is fully DNSSEC-signed; and (4) specifies a secure issuance policy or blocks wildcard issuance at the parent domain.

Specifying the accounturi parameter results in domain control validation that matches the description of the "known-account-specifier" protocol. Specifying the "dns-01" validationmethod results in validation per the "secure-dns-record-change" protocol if the site operator ensures that all DNS records for DV (in particular, the "_acme-challenge" subdomain, and any domains to which it is CNAMEd) are DNSSEC-signed. About 74.8K domains are using the elements of cryptographic DV via these mechanisms (Section 4.5).

Future work estimates. The CA estimates the effort of implementing the `security` CAA tag (post standardization) to be similar to RFC 8657 implementation. Recognizing and parsing the record is no more complex than parsing other CAA properties. Detecting whether the full DNS query was protected by DNSSEC (i.e., the AD bit is set in all responses) and passing that boolean into the CAA-checking routine requires work comparable to implementing the validationmethods parameter. Finally, the open source OpenMPIC project is also working to integrate both RFC 8657 and our `security tag` extension. OpenMPIC is used by multiple CAs including Sectigo and HARICA, and provides functionality to process CAA records from multiple vantage points on the internet.

5.2 Standardization Efforts

Standardization of our technique is essential for adoption. We focus our standardization efforts on two initiatives: (1) ensuring cryptographic DV's security properties by requiring all CAs to validate DNSSEC and (2) enabling domain owners to set strong, declarative security properties via our `security` tag.

DNSSEC Validation by CAs. As discussed in Section 6.1, *all* CAs must perform DNSSEC validation for CAA record queries for our posited security properties to hold. If an insecure CA does not validate DNSSEC for CAA records, an adversary could target the insecure CA to fraudulently obtain a valid cert.

The current Baseline Requirements of the CA/Browser Forum mention DNSSEC only once: which states that CAA lookups cannot fail open on DNSSEC-signed domains (in Section 3.2.2.8: CAA

Table 2: Result of empirical tests against top CAs. CAB means a CA properly block issuance if it fails to retrieve the CAA records of a DNSSEC-signed domain (per the CA/Browser Forum Baseline Requirements). DNSSEC means a CA validates DNSSEC ensuring it retrieves the legitimate CAA records for a DNSSEC-signed domain. Both of these properties together imply the CA will uphold the security of cryptographic DV.

CA	Test Case		"Secure" CA?
	CAB	DNSSEC	
Let's Encrypt	✓	✓	✓
Digicert (GeoTrust)	✓	✗	✗
GoDaddy	✓	✗	✗
Sectigo	✓	✓	✓
Google Trust Services	✓	✓	✓
Certainly	✓	✓	✓

Records [16]). However, this requirement does not obligate CAs to perform DNSSEC validation on CAA lookups, since a CA can comply by always failing closed (i.e., never treating lookup errors as permission to issue). Several CAs and root programs have confirmed this interpretation. Consequently, an attacker can supply DNSSEC-invalid responses that avoid lookup failures at non-validating CAs.

Empirically, we found that DNSSEC validation behavior by CAs varied. To determine which CAs already validate DNSSEC for CAA records, we performed a suite of empirical tests for top CAs (Table 2). We found that several major CAs like Let's Encrypt, Google Trust Services, and Certainly already perform DNSSEC validation for CAA records. However, we also found that this behavior is not universal, and CAs like GoDaddy and Digicert do not validate DNSSEC for CAA records. Sectigo recently implemented DNSSEC validation over the course of our experiments.

Standardization Experience and Impact at CA/Browser Forum. Because several CAs did not perform DNSSEC validation, we presented a proposal for mandatory DNSSEC validation to the Apple, Chrome, and Microsoft root programs in August 2024; all supported requiring CAs to validate DNSSEC. We then proposed a ballot in the Validation Subcommittee of the CA/Browser Forum's Server Certificate Working Group to mandate DNSSEC validation for both DV and CAA queries. CAs were overwhelmingly supportive of the ballot and contributed many interesting considerations which improved the ballot text over several months, as discussed in Appendix D. Our ballot, SC-085, was formally proposed by Clint Wilson (Apple) and endorsed by Wayne Thayer (Fastly), Dimitris Zacharopoulos (HARICA), and Ryan Dickson (Chrome). The Ballot passed successfully after Forum participants' vote in June 2025. CAs will now be required to validate DNSSEC (when present) starting March 2026. This ballot has thus directly enabled the first ever cryptographically-provable way for domains to gain Web PKI certificates which cannot be downgraded to plaintext channels even in the presence of non-participating CAs. Because the Web PKI is only as strong as its weakest link, CA-wide DNSSEC validation substantially improves the security of all DNSSEC-protected domains.

Standardization efforts for CAA security tag at IETF. We have written a specification for the CAA security tag, which has been submitted as a working draft to the IETF Limited Additional Mechanisms for PKIX and SMIME (LAMPS) working group after our presentation at IETF 122 in Bangkok. The specification is still undergoing discussion within the IETF working group. In our engagements so far, several IETF members have supported the idea of an explicit CAA tag for cryptographic domain control validation.

Future Directions for Standardization. We recommend that the CA/Browser Forum consider two concrete updates to the BRs: (1) standardizing support for RFC8657 CAA semantics, to avoid the failure mode we uncovered where non-participating CAs ignore granular policy specifications (Section 4.1.1); (2) incorporate our DV methods that provide declarative security with cryptographic checks (Section 4.2) with an associated IETF specification. Recent CA/Browser Forum efforts such as SC-082 Redux [59] highlight growing interest in more robust domain validation mechanisms, which aligns well with cryptographic DV.

6 Security Analysis

We verify the security of cryptographic DV through a combination of formal analysis and ethically conducted attacks. We develop a formal model using the Tamarin prover [58] to prove that we achieve (1) *secure issuance*: A CA supporting cryptographic DV only accepts certificate signing requests from the legitimate domain owner, and (2) *downgrade prevention*: A CA that does not support cryptographic DV but can securely look up domain owner policies, rejects any certificate signing requests for opt-in domains. We also demonstrate the resilience of cryptographic DV to ethically-launched real-world attacks. While more limited in scope than formal analysis, empirical testing verifies ability of our framework to protect a real-world CA (as opposed to an instantiated model) from fraudulent certificate issuances due to on-path or off-path attacks.

6.1 Formal Security Analysis

The goal of our formal analysis is to rigorously derive the specific security properties achieved by cryptographic DV. We focus on the security guarantees when using DNSSEC to securely lookup domain owner policies since it is fully specified by mature standards.

Tamarin model. We create a formal model using the security protocol verification tool Tamarin [58] based on the relevant protocol-specific standardization documents, such as IETF drafts and CA/Browser Forum documents [16]. Conceptually, our model consists of two phases: (1) the setup phase, where we define all actors (domain owners, CAs [16], and Dolev-Yao adversaries) and their cryptographic keys (TLS, DNSSEC, ACME [5]), the DNS(SEC) topology, and the intended certificate requests, and (2) the protocol phase, where we run all relevant protocol steps, such as fetching CAA [38, 53] and NSEC [4, 35, 55] records, performing domain control validation [16], and issuing X.509 certificates [22, 62]. The model then proves that for a given setup phase, which represents "ground truth", e.g., the domain owner uploads a DNSSEC-protected CAA security tag, certain guarantees will hold in the protocol phase, e.g., a CA does not issue a fake certificate. We ensure that the security properties of cryptographic DV hold in any scenario, by allowing the setup phase to create arbitrary DNS(SEC) topologies and certificate requests. The model is publicly available [52] and contains a README file with additional details.

Listing 1: Simplified Tamarin lemma proving that only the legitimate domain owner can request a certificate.

```
lemma SecureRecordChangeIssuanceSimplified:
 All domain ID #i #j.
   // If a secure CA accepts an owner ID
   // validated using secure-dns-record-change,
   DnsRecordChangeAccepted('secCA', domain, ID) @i
   // for a DNSSEC-protected CAA security tag,
   DnssecProtectedCaaSecurityTagExists(domain) @j
 ==>
   // then it was the domain owner's legitimate ID
   (Ex #k. IsLegitimateOwnerID(domain, ID) @k)
```

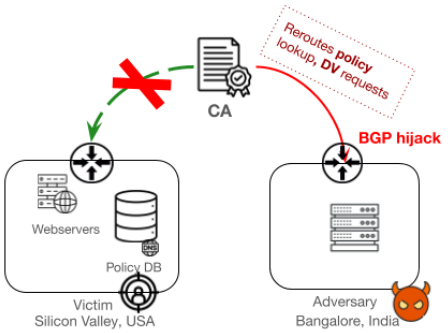


Figure 5: Setup for ethical attacks performed.

Assumptions. We assume that domain owners either specify an *ownerID* or a secure validation method, and protect the integrity of their DNS records through DNSSEC. Furthermore, since modeling the entirety of the DNS(SEC) protocol is beyond the scope of this work, we focus our model on the relevant protocol aspects. In particular, (1) our model considers the temporal ordering of events but does not consider concrete time intervals, e.g., validation result reuse at CAs of 398 days, (2) we change protocol-specific encodings to a Tamarin-compatible format and exclude auxiliary records and fields, e.g., the salt used in NSEC3 records, (3) we focus on the online KSK mode for DNSSEC, and (4) we treat revoked keys as non-existent keys. Section E provides additional details on our assumptions and Tamarin proof modeling.

Security Properties. We classify CAs based on two criteria. A "secure" CA properly validates DNSSEC signature chains on CAA records, and rejects issuance in case of a failed validation, while an "insecure" CA may skip or ignore validation results. Based on our empirical analysis shown in Section 5.2, at least three high volume CAs, namely Let's Encrypt, Google Trust Services, and Certainly, are already secure. Furthermore, all CAs will now transition to "secure" status as a result of our standardization effort that mandates DNSSEC validation of CAA and DV-related records. A "participating" CA correctly implements cryptographic DV and rejects issuance if it does not conform to the domain policy specified in the CAA record, while a "non-participating" CA is a legacy CA that does not implement cryptographic DV.

Property 1: Secure issuance. A secure and participating CA only accepts certificate signing requests from the legitimate domain owner. An implication of this property is that if the root store consists of secure and participating CAs, global hijacking attacks against domain owners that specify a *ownerID* or a secure validation method in their CAA records, are rendered ineffective.

Property 2: Downgrade prevention. *If the critical bit in the CAA records is set*, a secure and non-participating CA rejects any certificate signing requests for opt-in domains. An implication of this and the previous property is that if the root store consists of secure (participating or non-participating) CAs, global hijacking attacks against domains that specify an *ownerID* or a secure DV method in their CAA records and *set the critical flag*, are rendered ineffective.

Proofs. In Tamarin, properties are proven through lemmas, which are logical statements over possible protocol executions, called protocol traces. Concretely, our security properties are proven by lemma `KnownAccountSpecifierIssuance` if the domain owner specifies an *ownerID* and by lemma `SecureRecordChangeIssuance` if the domain owner specifies the secure validation method secure-dns-record-change. A simplified version of the second lemma is shown in Listing 1: In all protocol traces where an *ownerID* is accepted by the secure CA to issue a certificate, the *ownerID* either belongs to the legitimate domain owner, or the domain owner revealed its DNSSEC key allowing an adversary to forge a CAA record. In addition to proving protocol properties, we ensure that our model is executable, i.e., the model allows for successful certificate issuance and validation, as well as attacks if certain assumptions do not hold. For example, lemma `FakeCertificateWithoutCriticalFlag` shows that if the domain owner does not set the critical flag in their CAA security record, a non-participating secure CA will issue a fake certificate. Through a combination of such lemmas, we cover our security properties and prove that, given the threat model described in Section 3.1, all security properties hold under the stated assumptions.

6.2 Empirical Security Analysis

Next, we demonstrate the security of our approach by ethically conducting network attacks, enumerated in Table 1. These attacks mirror the edge cases and confirm the vulnerabilities discussed in Section 4.1.1. We empirically show that while current conventional DV is indeed vulnerable to both off-path and on-path attacks, cryptographic DV *blocks malicious certificate issuance in all cases.*

Attack Setup. Our attack setup uses a victim domain hosted on a web server and DNS nameserver in a Silicon Valley cloud datacenter as shown in Fig. 5. We evaluate two attacker regimes: (1) an *on-path* adversary, modeled as a reverse proxy situated on path between the CA and victim's web and DNS servers; (2) an *off-path* adversary with BGP-announcement capability, that independently sub-prefix hijacks the prefixes hosting the victim web and DNS servers. We test both a "secure" opt-in CA that validates DNSSEC for CAA records and implements our cryptographic DV extensions, and an "insecure" CA which does not validate DNSSEC for CAA or DV retrievals. We stress that the attacker's location is largely irrelevant for the takeaways of these attacks. For the off-path case, sub-prefix BGP hijacks have global reach; for on-path, any attacker on the

Table 3: Empirical results of network attacks trying to gain a certificate for a victim domain against a secure CA.

Attack	Certificate issued?
Secure domain policies	
Denial of Service on policy retrieval (Timeout for CAA lookup)	No
Forged domain policy (missing DNSSEC Signature on CAA in signed zone)	No
Forged domain policy (invalid DNSSEC signature on CAA record)	No
Attacks on DV challenge	
Forged Presence of HTTP Challenge	No
Forged DNS record for DV Challenge	No

path between the victim's infrastructure and the CA can perform attacks to similar effect. Experiments used a customized BIND 9 server and the Python library scapy to manipulate responses for domain policy lookups and DNS-based DV challenges.

Ethical Considerations. Our attacks are conducted ethically via the following principles. The network attacks caused no disruption to real-world users or services. The experiments exclusively used domain names, web servers, and DNS nameservers under our control, and all BGP announcements targeted IP prefixes owned by us. Finally, no real real users were affected by our tests, as all test prefixes and domains hosted no real-world services.

Results. Table 3 describes the outcomes of a suite of network attacks against a test victim domain by an attacker requesting a certificate from a real-world CA. The victim domain registers a DNSSEC-signed CAA record specifying an ACME-compliant *ownerID* in an issue tag for a secure, participating CA. For *all attacks performed*, the secure CA denied the attacker's certificate request and blocked the attack. While any secure but non-participating CA will correctly block malicious certificate requests (such as those in our tests), we also confirmed that a secure, participating CA does allow the legitimate domain owner to obtain a certificate. Finally, we tested attacks on domain policy retrieval and DNS-based DV for an insecure (i.e., non-DNSSEC-validating) CA. We found that the adversary could successfully obtain a certificate from an insecure CA by forging the CAA policy. Our standardization efforts to mandate DNSSEC validation for all DV lookups will transition all CAs to "secure" CAs. Overall, our empirical analysis shows that the security properties "secure issuance" and "downgrade prevention" proved via formal analysis on a system model hold for real-world deployment.

7 Related Work

We discuss related work beyond technologies covered in Section 2.

Attacks and defenses on domain validation. An extensive body of work shows that weaknesses in BGP [9, 10, 13, 19, 25, 34] and DNS(SEC) [12, 24, 39] can be exploited to fraudulently obtain valid TLS certificates. Localized BGP hijacks can be mitigated by multi-perspective probing [1, 73], e.g., in Multi-Perspective-Issuance Corroboration (MPIC) where CAs perform domain validation from multiple vantage points [10, 12]. The Resource PKI (RPKI) and Route origin Validation (ROV) offer limited protection against BGP hijack attacks [36, 41, 42, 60, 61, 76], while BGPsec [57] and SCION [18] completely mitigate hijacks but are not widely deployed. Cryptographic DV goes beyond this threat model and offers protection against *all network adversaries*, including those on-path.

Security beyond domain validation. An important body of work addresses conceptual weaknesses in the Web PKI, such as lack of accountability and the risk of compromised CAs. A prominent example is Certificate Transparency (CT) [54], which enables detection of misissued certificates via append-only public logs. However, unlike cryptographic DV, CT does not prevent malicious certificates from being issued in the first place.

Similar to DANE and HPKP (Section 2), PoliCert [71], DTKI [77], and F-PKI [17] propose policy mechanisms that allow domain owners to restrict the validity of their TLS certificates. Unlike our approach, these systems target *client devices* (e.g., an end-user using a web browser), not the CAs. Their threat models are often stricter than the threat model of cryptographic DV since they can tolerate some malicious CAs while still providing their respective security guarantees. However, they ultimately rely on benign CAs validating domain ownership, and are thus orthogonal to cryptographic DV which protects domain validation against network adversaries.

AKI [50] and ARPKI [6] address the Web PKI's weakest-link security issue, originating from the oligopoly CA model. By involving multiple entities in the certificate issuance process, they ensure that even if multiple parties, e.g., CAs, are compromised, an adversary is not able to generate a fake certificate. Similarly, PoliCert [71] and Cothority [68] employ multi-signature schemes to issue and protect the domain policies and TLS certificates, respectively.

NOPE [26] is a recent proposal for including small zero-knowledge proofs in Web PKI certificates that attest to the existence of a DNSSEC-signed record containing the certificate's public key at the time of issuance. However, NOPE is susceptible to downgrade attacks on initial non-pinned TLS connections and requires webserver-side software changes. Cryptographic DV's declarative security enables the easy integration of DNSSEC alternatives, such as RHINE [30] which fundamentally improve DNSSEC's security guarantees by providing authority independence of (sub-)zones.

Finally, in the context of the Tor network, onion services cryptographically connect their address to their public keys, enabling CA-independent self-authentification [27]. However, this raises a usability challenge for service discovery [74], and overcoming this challenge has been the subject of several recent works [23, 48, 69, 70].

8 Conclusion

We advocate a vision for securing the central building block of the Web PKI, the domain validation protocol, via cryptographic guarantees. Our work takes an important step toward addressing a long-standing HTTPS bootstrapping problem: while HTTPS itself protects against network adversaries, the Web PKI introduces vulnerabilities in how trust parameters are established. By using domain owner policies and `authenticated DNS`, our work addresses the weaknesses in existing DV procedures to enable cryptographic verification. This ultimately enables HTTPS-based privacy and integrity that is resilient to network adversaries.

Our approach can be easily deployed: it requires no changes to client-side software and provides security benefits with a single participating CA. Our only required global change, CAs' usage of

secure DNS lookup mechanisms, has already been standardized at the CA/Browser Forum. Finally, we view this work as a first step towards true CA accountability, as authenticated domain control validation can ultimately be recorded in CT logs and serve as proof of a CA's issuance authorization.

Acknowledgments

We thank Let's Encrypt engineers for their feedback on the design of cryptographic DV, for sharing their experiences on implementing sophisticated CAA-based policy parsing, and for sharing operational logs that enabled our deployability analysis. Our standardization impact at the CA/Browser forum would not have been possible without the support of Clint Wilson, Ryan Dickson, Chris Clements, and Wayne Thayer. We are grateful to Josh Aas, Richard Barnes, Adrian Perrig, and Jennifer Rexford for providing helpful feedback on our work. This material is based on work supported by the Defense Advanced Research Project Agency (DARPA) under contract no. HR00112590081, the Open Technology Foundation, and Princeton's Center for Information Technology Policy. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors, and do not necessarily reflect the views of the sponsors.

References

[1] Mansoor Alicherry and Angelos D. Keromytis. 2009. DoubleCheck: Multi-path verification against man-in-the-middle attacks. In *Proceedings of the IEEE Symposium on Computers and Communications (ISCC)*. https://doi.org/10.1109/iscc.2009.5202224

[2] Apple. 2023. Apple's Certificate Transparency log program. https://support.apple.com/en-us/103703.

[3] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. 2005. *Protocol Modifications for the DNS Security Extensions.* RFC 4035. IETF. http://tools.ietf.org/rfc/rfc4035.txt

[4] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. 2005. *Resource Records for the DNS Security Extensions.* RFC 4034. IETF. http://tools.ietf.org/rfc/rfc4034.txt

[5] R. Barnes, J. Hoffman-Andrews, D. McCarney, and J. Kasten. 2019. *Automatic Certificate Management Environment (ACME).* RFC 8555. IETF. http://tools.ietf.org/rfc/rfc8555.txt

[6] David Basin, Cas Cremers, Tiffany Hyun-Jin Kim, Adrian Perrig, Ralf Sasse, and Pawel Szalachowski. 2014. ARPKI: Attack Resilient Public-Key Infrastructure. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. https://doi.org/10.1145/2660267.2660298

[7] Karthikeyan Bhargavan, Abhishek Bichhawat, Quoc Huy Do, Pedram Hosseyni, Ralf Küsters, Guido Schmitz, and Tim Würtele. 2021. An In-Depth Symbolic Security Analysis of the ACME Standard. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. https://doi.org/10.1145/3460120.3484588

[8] Henry Birge-Lee. 2022. Attackers exploit fundamental flaw in the web's security to steal $2 million in cryptocurrency. https://freedom-to-tinker.com/2022/03/09/attackers-exploit-fundamental-flaw-in-the-webs-security-to-steal-2-million-in-cryptocurrency/.

[9] Henry Birge-Lee, Yixin Sun, Anne Edmundson, Jennifer Rexford, and Prateek Mittal. 2018. Bamboozling Certificate Authorities with BGP. In *Proceedings of the USENIX Security Symposium*. 833–849.

[10] Henry Birge-Lee, Liang Wang, Daniel McCarney, Roland Shoemaker, Jennifer Rexford, and Prateek Mittal. 2021. Experiences Deploying Multi-Vantage-Point Domain Validation at Let's Encrypt. In *Proceedings of the USENIX Security Symposium*. 4311–4327.

[11] Kevin Borgolte, Tobias Fiebig, Shuang Hao, Christopher Kruegel, and Giovanni Vigna. 2018. Cloud Strife: Mitigating the Security Risks of Domain-Validated Certificates. In *Proceedings of the Applied Networking Research Workshop (ANRW)*. https://doi.org/10.1145/3232755.3232859

[12] Markus Brandt, Tianxiang Dai, Amit Klein, Haya Shulman, and Michael Waidner. 2018. Domain Validation++ For MitM-Resilient PKI. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. 2060–2076. https://doi.org/10.1145/3243734.3243790

[13] Markus Brandt, Haya Shulman, and Michael Waidner. 2021. Evaluating Resilience of Domains in PKI. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. https://doi.org/10.1145/3460120.3485356

[14] CA/Browser Forum. 2024. Ballot SC-67 v3: Require domain validation and CAA checks to be performed from multiple Network Perspectives Corroboration. https://cabforum.org/2024/08/05/ballot-sc-67-v3-require-domain-validation-and-caa-checks-to-be-performed-from-multiple-network-perspectives-corroboration/.

[15] CA/Browser Forum 2024. *Bylaws of the CA/Browser Forum.* Version 2.5. CA/Browser Forum.

[16] CA/Browser Forum 2025. *Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates.* Version 2.1.5. CA/Browser Forum.

[17] Laurent Chuat, Cyrill Krähenbühl, Prateek Mittal, and Adrian Perrig. 2022. F-PKI: Enabling Innovation and Trust Flexibility in the HTTPS Public-Key Infrastructure. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS)*. https://doi.org/10.14722/ndss.2022.24241

[18] Laurent Chuat, Markus Legner, David Basin, David Hausheer, Samuel Hitz, Peter Müller, and Adrian Perrig. 2022. *The Complete Guide to SCION.* Springer International Publishing. https://doi.org/10.1007/978-3-031-05288-0

[19] Grace H. Cimaszewski, Henry Birge-Lee, Liang Wang, Jennifer Rexford, and Prateek Mittal. 2023. How Effective is Multiple-Vantage-Point Domain Control Validation?. In *Proceedings of the USENIX Security Symposium*. 5701–5718. https://www.usenix.org/conference/usenixsecurity23/presentation/cimaszewski

[20] Cloudflare. 2024. Encrypt DNS Traffic. https://developers.cloudflare.com/1.1.1.1/encryption/.

[21] Cloudflare. 2025. Delegated DCV — Domain Control Validation. https://developers.cloudflare.com/ssl/edge-certificates/changing-dcv-method/methods/delegated-dcv/ Accessed on 2025-05-30.

[22] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. 2008. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.* RFC 5280. IETF. http://tools.ietf.org/rfc/rfc5280.txt

[23] Rasmus Dahlberg, Paul Syverson, Linus Nordberg, and Matthew Finkel. 2022. Sauteed Onions: Transparent Associations from Domain Names to Onion Addresses. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society (WPES)*. 35–40. https://doi.org/10.1145/3559613.3563208

[24] Tianxiang Dai, Philipp Jeitner, Haya Shulman, and Michael Waidner. 2021. The Hijackers Guide To The Galaxy: Off-Path Taking Over Internet Resources. In *Proceedings of the USENIX Security Symposium*. 3147–3164. https://www.usenix.org/conference/usenixsecurity21/presentation/dai

[25] Tianxiang Dai, Haya Shulman, and Michael Waidner. 2021. Let's Downgrade Let's Encrypt. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*.

[26] Zachary DeStefano, Jeff J. Ma, Joseph Bonneau, and Michael Walfish. 2024. NOPE: Strengthening domain authentication with succinct proofs. In *Proceedings of the ACM SIGOPS Symposium on Operating Systems Principles (SOSP)*. 673–692. https://doi.org/10.1145/3694715.3695962

[27] Roger Dingledine, Nick Mathewson, Paul F Syverson, et al. 2004. Tor: The second-generation onion router.. In *USENIX security symposium*, Vol. 4. 303–320.

[28] Google Public DNS. 2024. Secure transports for DNS. https://developers.google.com/speed/public-dns/docs/secure-transports.

[29] DNS Flag Day. 2020. 2020 | DNS flag day. https://www.dnsflagday.net/2020/.

[30] Huayi Duan, Rubén Fischer, Jie Lou, Si Liu, David Basin, and Adrian Perrig. 2023. RHINE: Robust and High-performance Internet Naming with E2E Authenticity. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*.

[31] V. Dukhovni and W. Hardaker. 2015. *The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance.* RFC 7671. IETF. http://tools.ietf.org/rfc/rfc7671.txt

[32] C. Evans, C. Palmer, and R. Sleevi. 2015. *Public Key Pinning Extension for HTTP.* RFC 7469. IETF. http://tools.ietf.org/rfc/rfc7469.txt

[33] Fastly. 2025. Setting up TLS with certificates Fastly manages. https://www.fastly.com/documentation/guides/getting-started/domains/securing-domains/setting-up-tls-with-certificates-fastly-manages/ Accessed on 2025-05-30.

[34] Artyom Gavrichenkov. 2015. Breaking HTTPS with BGP Hijacking. In *BlackHat USA*.

[35] R. Gieben and W. Mekking. 2014. *Authenticated Denial of Existence in the DNS.* RFC 7129. IETF. http://tools.ietf.org/rfc/rfc7129.txt

[36] Yossi Gilad, Avichai Cohen, Amir Herzberg, Michael Schapira, and Haya Shulman. 2017. Are We There Yet? On RPKI's Deployment and Security. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS)*. https://doi.org/10.14722/ndss.2017.23123

[37] P. Hallam-Baker and R. Stradling. 2013. *DNS Certification Authority Authorization (CAA) Resource Record.* RFC 6844. IETF. http://tools.ietf.org/rfc/rfc6844.txt

[38] P. Hallam-Baker, R. Stradling, and J. Hoffman-Andrews. 2019. *DNS Certification Authority Authorization (CAA) Resource Record.* RFC 8659. IETF. http://tools.ietf.org/rfc/rfc8659.txt

[39] Elias Heftrig, Haya Shulman, and Michael Waidner. 2023. Downgrading DNSSEC: How to Exploit Crypto Agility for Hijacking Signed Zones. In *Proceedings of the USENIX Security Symposium*. 7429–7444. https://www.usenix.org/conference/us

enixsecurity23/presentation/heftrig
[40] Scott Helme. 2020. HPKP is no more! https://scotthelme.co.uk/hpkp-is-no-more/.
[41] Tomas Hlavacek, Philipp Jeitner, Donika Mirdita, Haya Shulman, and Michael Waidner. 2022. Stalloris: RPKI Downgrade Attack. In *Proceedings of the USENIX Security Symposium*. 4455–4471. https://www.usenix.org/conference/usenixsecurity22/presentation/hlavacek
[42] Tomas Hlavacek, Haya Shulman, Niklas Vogel, and Michael Waidner. 2023. Keep Your Friends Close, but Your Routeservers Closer: Insights into RPKI Validation in the Internet. In *Proceedings of the USENIX Security Symposium*. 4841–4858. https://www.usenix.org/conference/usenixsecurity23/presentation/hlavacek
[43] P. Hoffman. 2023. *DNS Security Extensions (DNSSEC)*. RFC 9364. IETF. http://tools.ietf.org/rfc/rfc9364.txt
[44] P. Hoffman and J. Schlyter. 2012. *The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA*. RFC 6698. IETF. http://tools.ietf.org/rfc/rfc6698.txt
[45] Hans Hoogstraaten, Ronald Prins, Daniël Niggebrugge, Danny Heppener, Frank Groenewegen, Janna Wettink, Kevin Strooy, Pascal Arends, Paul Pols, Robbert Kouprie, Steffen Moorrees, Xander van Pelt, and Yun Zheng Hu. 2012. *Black Tulip: Report of the investigation into the DigiNotar Certificate Authority breach*. Technical Report. The Hague University of Applied Sciences. https://doi.org/10.13140/2.1.2456.7364
[46] Geoff Huston. 2019. Some bad news for DANE (and DNSSEC). https://blog.apnic.net/2019/05/27/dns-oarc-30-bad-news-for-dane/.
[47] Geoff Huston. 2023. Measuring the use of DNSSEC. https://blog.apnic.net/2023/09/18/measuring-the-use-of-dnssec/.
[48] Aaron D. Jaggard, Paul Syverson, and Catherine Meadows. 2024. A Logic of Sattestation. In *IEEE Computer Security Foundations Symposium (CSF)*. 356–371. https://doi.org/10.1109/CSF61375.2024.00047
[49] Peter Kacherginsky. 2022. Celer Bridge incident analysis. https://www.coinbase.com/blog/celer-bridge-incident-analysis.
[50] Tiffany Hyun-Jin Kim, Lin-Shung Huang, Adrian Perrig, Collin Jackson, and Virgil Gligor. 2013. Accountable key infrastructure (AKI): A Proposal for a Public-Key Validation Infrastructure. In *Proceedings of the International Conference on World Wide Web*. https://doi.org/10.1145/2488388.2488448
[51] O. Kolkman, W. Mekking, and R. Gieben. 2012. *DNSSEC Operational Practices, Version 2*. RFC 6781. IETF. http://tools.ietf.org/rfc/rfc6781.txt
[52] Cyrill Krähenbühl. 2025. Cryptographic DV Tamarin Model. https://github.com/inspire-group/cryptographic-dv-tamarin-model/tree/pets-2026.
[53] H. Landau. 2019. *Certification Authority Authorization (CAA) Record Extensions for Account URI and Automatic Certificate Management Environment (ACME) Method Binding*. RFC 8657. IETF. http://tools.ietf.org/rfc/rfc8657.txt
[54] B. Laurie, A. Langley, and E. Kasper. 2013. *Certificate Transparency*. RFC 6962. IETF. http://tools.ietf.org/rfc/rfc6962.txt
[55] B. Laurie, G. Sisson, R. Arends, and D. Blacka. 2008. *DNS Security (DNSSEC) Hashed Authenticated Denial of Existence*. RFC 5155. IETF. http://tools.ietf.org/rfc/rfc5155.txt
[56] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2019. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS)*. https://doi.org/10.14722/ndss.2019.23386
[57] M. Lepinski and K. Sriram. 2017. *BGPsec Protocol Specification*. RFC 8205. IETF. http://tools.ietf.org/rfc/rfc8205.txt
[58] Simon Meier, Benedikt Schmidt, Cas Cremers, and David Basin. 2013. The TAMARIN Prover for the Symbolic Analysis of Security Protocols. In *Computer Aided Verification*. Springer Berlin Heidelberg, 696–701. https://doi.org/10.1007/978-3-642-39799-8_48
[59] Michael Slaughter. 2025. slghtr-says/servercert at SC_082_redux. https://github.com/slghtr-says/servercert/tree/SC_082_redux.
[60] Donika Mirdita, Haya Schulmann, and Michael Waidner. 2024. SoK: An Introspective Analysis of RPKI Security. https://doi.org/10.48550/ARXIV.2408.12359
[61] National Institute of Standards and Technology (NIST). 2024. NIST RPKI Monitor. https://rpki-monitor.antd.nist.gov/.
[62] M. Nystrom and B. Kaliski. 2000. *PKCS #10: Certification Request Syntax Specification Version 1.7*. RFC 2986. IETF. http://tools.ietf.org/rfc/rfc2986.txt
[63] D. Quigley, J. Lu, and T. Haynes. 2015. *Registry Specification for Mandatory Access Control (MAC) Security Label Formats*. RFC 7569. IETF. http://tools.ietf.org/rfc/rfc7569.txt
[64] Shivan Sahib. 2023. Delegated domain verification. https://blog.apnic.net/2023/07/03/delegated-domain-verification/.
[65] B. Schwartz, M. Bishop, and E. Nygren. 2023. *Service Binding and Parameter Specification via the DNS (SVCB and HTTPS Resource Records)*. RFC 9460. RFC Editor.
[66] Sectigo. 2025. Sectigo TLS Certificates Certificate Policy and Certification Practice Statement. https://www.sectigo.com/uploads/files/Sectigo_TLS_CPS_v6_1_0.pdf.
[67] R.B. Shoemaker. 2020. *Automated Certificate Management Environment (ACME) TLS Application-Layer Protocol Negotiation (ALPN) Challenge Extension*. RFC 8737. IETF. http://tools.ietf.org/rfc/rfc8737.txt
[68] Ewa Syta, Iulia Tamas, Dylan Visher, David Isaac Wolinsky, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ismail Khoffi, and Bryan Ford. 2016. Keeping Authorities "Honest or Bust" with Decentralized Witness Cosigning. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*. https://doi.org/10.1109/sp.2016.38
[69] Paul Syverson, Matthew Finkel, Saba Eskandarian, and Dan Boneh. 2021. Attacks on Onion Discovery and Remedies via Self-Authenticating Traditional Addresses. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society (WPES)*. 45–52. https://doi.org/10.1145/3463676.3485610
[70] Paul Syverson and Matthew Traudt. 2019. Self-Authenticating Traditional Domain Names. In *IEEE Secure Development (SecDev)*. 147–160. https://doi.org/10.1109/SecDev.2019.00030
[71] Pawel Szalachowski, Stephanos Matsumoto, and Adrian Perrig. 2014. PoliCert: Secure and Flexible TLS Certificate Management. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. https://doi.org/10.1145/2660267.2660355
[72] tojens. 2021. Making DoH Discoverable: Introducing DDR. https://techcommunity.microsoft.com/blog/networkingblog/making-doh-discoverable-introducing-ddr/2887289.
[73] Dan Wendlandt, David G. Andersen, and Adrian Perrig. 2008. Perspectives: Improving SSH-style Host Authentication with Multi-Path Probing. In *Proceedings of the USENIX Annual Technical Conference (ATC)*.
[74] Philipp Winter, Anne Edmundson, Laura M. Roberts, Agnieszka Dutkowska-Żuk, Marshini Chetty, and Nick Feamster. 2018. How Do Tor Users Interact With Onion Services?. In *Proceedings of the USENIX Security Symposium*. 411–428. https://www.usenix.org/conference/usenixsecurity18/presentation/winter
[75] Sara Wrótniak, Hemi Leibowitz, Ewa Syta, and Amir Herzberg. 2019. Provable Security for PKI Schemes. Cryptology ePrint Archive, Paper 2019/807. https://eprint.iacr.org/2019/807/20240514:190528
[76] Matthias Wählisch, Robert Schmidt, Thomas C. Schmidt, Olaf Maennel, Steve Uhlig, and Gareth Tyson. 2015. RiPKI: The Tragic Story of RPKI Deployment in the Web Ecosystem. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, Vol. 4. 1–7. https://doi.org/10.1145/2834050.2834102
[77] Jiangshan Yu, Vincent Cheval, and Mark Ryan. 2016. DTKI: A New Formalized PKI with Verifiable Trusted Parties. *Comput. J.* 59, 11 (July 2016), 1695–1713. https://doi.org/10.1093/comjnl/bxw039

A Required Properties of Secure-Channel DNS

First, the domain owner must ensure that *only secure nameservers* are used, i.e., nameservers offering secure channels. Otherwise, an adversary may perform denial-of-service attacks on secure nameservers and trick the CA's DNS resolver to use an insecure nameserver without support for secure channels to spoof DNS records. Second, the secure channel approach must *prevent downgrade attacks*, where an adversary convinces the CA's DNS resolver to fall back to a less secure DNS transport method with a secure nameserver, e.g., plaintext DNS lookup. Third, the secure channels to the relevant nameservers must be used by *all* CAs. This is required to prevent an adversary from approaching a less secure CA that did not implement this secure channels.

Additionally, the CA/Browser Forum Baseline Requirements for TLS server certificate issuance [16] stipulate that failure to retrieve the CAA record of a DNSSEC-protected domain must block issuance by a CA. We leverage this property to protect against adversaries launching DoS attacks on domain policies. For secure channel DNS to achieve the same security properties as DNSSEC in the context of cryptographic DV, the Baseline Requirements need to be amended to also stipulate that the failure of a CAA DNS lookup performed over a secure channel must similarly block issuance.

As of today, DNSSEC is more supported by both recursive resolvers (several CA recursive resolvers validate DNSSEC, see Table 2) and authoritative nameservers (many authoritative DNS providers offer DNSSEC). However, with the large degree of centralization in the DNS hosting of PKI domains [19], a significant portion of domains could benefit from `authenticated DNS lookups` via

Table 4: Impact of critical flag.

Deployment Level	Prevent attacks	
	with flag	w/o flag
Cryptographic DV (all CAs)	✓	✓
Secure CAA lookup (all CAs)	✓	✗
Secure CAA lookup (some CAs)	✗	✗

secure channels with only a small group of opt-in participants. Furthermore, several large DNS providers already support secure DNS channels for client-to-recursive queries [20, 28]

A.1 Instantiation of Secure DNS Channels Using SVCB Records

One practical way to bootstrap the establishment of secure channels between recursive and authoritative nameservers is by creating SVCB DNS records [65] associated with authoritative nameservers. Using SVCB records, a nameserver can specify various attributes for entities connecting to it (e.g., an Application-Layer Protocol Name or an Encrypted Client Helo Key). In fact, Microsoft already uses SVCB records for DoH discovery [72]. Information on the cryptographic keys used during the establishment of secure tunnels could also potentially be contained in SVCB records. By securing these SVCB records using DNSSEC instead of Web PKI certificates, we break the circular dependency where a CA requires an authentic Web PKI certificate to securely perform domain validation in order to issue new Web PKI certificates. Hence, by leveraging DNSSEC deployment for the nameservers of a small number of widely-used DNS providers, we enable authenticated retrieval of CAA records for a large fraction of domains. This scales significantly better than the traditional DNSSEC deployment model and only requires participation in DNSSEC by nameservers, not individual domains.

A.2 Implications of DNS provider centralization

Prior work has demonstrated that DNS hosting is concentrated in a handful of major commercial providers [19]. Building upon these results, we observe that establishing secure tunnels to a small number (as few as 7) of DNS providers can authenticate lookups for a large segment of domains that may not otherwise use DNSSEC (as many as 60%+). These results show the feasibility of `authenticated DNS` via secure channels. In fact, Cloudflare, which hosts the highest percent of domains, already supports DoH on some of its DNS endpoints. A major anonymous CA is working to incorporate our proposed `authenticated DNS` via secure channels. Given that DNSSEC deployment hovers around 5% and Cloudflare alone hosts a factor of 3-4x more domains, even an initial deployment with a single DoH tunnel to Cloudflare would vastly expand the number of domains that support `authenticated DNS`.

B Example of Differences Between Cryptographic Verification and Traditional DV

The ACME "http-01" challenge strictly specifies use of the HTTP protocol over port 80 [5]. This makes performing a true "http-01" challenge over TLS difficult (only domains that redirect HTTP to HTTPS at the same label could potentially satisfy both requirements). Our proposed decoupling avoids this challenge by allowing a CA to perform the standard "http-01" challenge and then perform cryptographic identity verification via a second connection to the domain over port 443 via HTTPS.

C Confirming the Fail Open Behavior of CAA "issue" Tag Extensions

The CAA "`issue`" tag can be extended to support cryptographic DV using extensions in RFC 8657 [53]. However, any extensions to the CAA "`issue`" tag pose a serious security risk as they are interpreted solely at the discretion of the issuer per its Certificate Policy Statement (CPS) and have no formal mandate by the CA/Browser Forum.

We confirmed two cases where this behavior can hinder security by ethically requesting certificates for domains we controlled with various CAA records. The language used by the CAA RFC [38] and the CA/Browser Forum allows a CA to proceed with issuance even in the presence of a CAA record extension it does not understand (for the `issue` tag). This has two implications: 1) CAs will fail open on any newly proposed CAA extension for the `issue` tag and 2) even existing extensions like RFC 8657 will silently fail when used at a CA that does not support them.

Demonstrating certificate issuance with unknown CAA extensions for the `issue` tag. We used Let's Encrypt to test the behavior of a CA with an unknown CAA extension. We configured our test domain with the following CAA record: CAA 0 issue "letsencrypt.org; foosecurity=bar". We were able to obtain a certificate from Let's Encrypt for this domain even though Let's Encrypt did not understand the foosecurity extension. This is also supported by review of the relevant lines in Let's Encrypt source code seen here.[1]

Demonstrating certificate issuance with non-participating CAs for RFC 8657 extension. Next, we requested a certificate from Sectigo (via GoGetSSL) to test the behavior of a CA that does not support RFC 8657 when presented with RFC 8657 extensions. We configured our domain with the CAA record: CAA 0 issue "sectigo.com; validationmethods=dns-01". We then requested a certificate with *HTTP* domain control validation from Sectigo. Our certificate was issued despite it being a violation of the RFC 8657 extensions. To clarify, we do not find this event to be a violation of Sectigo's Certificate Policy Statement (v6.1.0) [66] or the CA/Browser Forum's Baseline Requirements (v2.1.5) [16] as neither of these documents in their current versions reference RFC 8657.

Even if the CA/Browser Forum requires all CAs to support RFC 8657 in the future (as we recommend in Section 5.2),unknown issue tag extensions will still fail open even under this ballot. Interestingly, we did find some precedent for failing closed on unknown issue tag extensions. ZeroSSL (which does use Sectigo to sign certificates) performs a "pre-flight check" to ensure domains are configured correctly before sending orders to Sectigo. We found this check to always fail when an unknown issue tag extension was present.

[1] Our CAA records had the critical bit unset, but the Let's Encrypt source code would interpret this record identically even if the critical bit was set, given the use of the `issue tag`.

An interesting avenue for future research is to reexamine the fail-open behavior of the CAA RFC 8659 [38] and devise a mechanism for critical/non critical distinction for issue tag extensions (e.g., CAs need to fail closed if an issue tag extension starts with "critical-").

D Working with the CA/Browser Forum on DNSSEC Validation

CA/Browser Forum members provided valuable feedback on our DNSSEC validation ballot, which helped to improve the ballot and formally move it to the public discussion phase. Below, we summarize feedback received during discussion in the Validation Subcommittee of the Server Certificate Working Group:

Maximizing the security Benefit: Mandating DNSSEC validation during CAA *and* DV. Our team's original proposal included wording to only mandate DNSSEC validation on CAA DNS record queries. This was the minimal change we needed to prevent downgrade attacks targeting non-participating CAs. Building upon this, forum members generally seemed to prefer the idea of a single ballot that mandated DNSSEC validation on both DV and CAA DNS queries because 1) additionally mandating DNSSEC validation during DV was deemed to have a minimal impact on the workload required to comply with the ballot and 2) mandating DNSSEC on DV queries provided an additional security benefit of reducing the attack surface of DNS attacks on DV. Since the increase in implementation burden was minimal and there was a noticable security improvement for mandating DNSSEC validation on DV as well, we revised the ballot to include DNSSEC validation on DV-related DNS queries at the encouragement of Forum members.

Creating a minimum viable standard: Referencing *sections for security mechanisms* in RFCs instead of entire documents. To optimize compliance and easy of verification, we chose to avoid introducing any new custom and ad-hoc language around how DNSSEC should be validated and instead mandated DNSSEC validation through the use of existing IETF RFCs. While this aids in compliance (and reduces the risk of a conflicting standard with an existing IETF RFC), many RFCs regarding DNSSEC discussed DNS behavior significantly beyond what was actually required to validate DNSSEC. This was particularly problematic due to language in RFCs which provided now obsolete operational guidance (e.g., RFC 4035 Section 4.1's recommendation that a security-aware resolver "SHOULD support a message size of 4000 octets" [3] which is now known to increase the vulnerability of DNS resolvers to fragmentation attacks [29]).

To avoid this, we cited algorithms defined in sections of the relevant RFCs (e.g., RFC 4035 Section 5) which avoided any unneeded operational considerations. Notably, RFCs are typically written to provide specific operational recommendations to best ensure compliance in the ecosystem. In contrast, the CA/Browser Forum Baseline Requirements are aimed at providing a minimum baseline that, if violated, has serious legal and financial implications. Thus, we (and Forum members) reviewed any content being pulled into the CA/Browser Forum Baseline Requirements from RFCs to ensure it provides only security-critical elements and contains as little operational guidance as possible which may become obsolete over time.

Understanding operational concerns: Providing a reasonable implementation timeline. One central question in the development of the ballot was to decide a concrete date when the DNSSEC validation requirement becomes effective. Our initial ballot targeted an effective date of November 15, 2025, but delaying this effective date allowed the ballot to gain significantly more support. Several CAs expressed concern over the timeline required to find and evaluate alternative DNS software in production settings. CAs are also forbidden from outsourcing any DNS operations used for DV or CAA lookups. Forum members stated that for large enterprises, the software procurement process could take a significant portion of the originally-proposed implementation time. Furthermore, feedback from CAs that had implemented DNSSEC validation (and Sectigo which recently enabled DNSSEC validation) showed that the largest operational risk came from misconfigured customer domains. To provide sufficient time to notify customers with DNSSEC errors and to provide time for sourcing of DNS software vendors, the effective date in the final ballot was pushed to March 15th, 2026 (which is one of the CA/Browser Forum's preferred dates for the implementation of new features). Overall, our experience indicates that Forum members are more open to new proposals from the research community if they are implemented with sufficient timelines to ensure smooth adoption.

We hope our experiences above provide helpful takeaways to any researchers interested in proposing new industry standards.

E Formal Model Details

We verify the security properties of our design, described in Section 6.1, by specifying the necessary Tamarin lemmas to prove our security properties and by modeling the behavior and capabilities of relevant entities as outlined below.

E.1 Assumptions

Our formal model design as well as the symbolic prover Tamarin itself lead to several assumptions on the adversary, (victim) domain owners, and their interaction with other relevant entities.

E.1.1 Adversary Capabilities. The adversary is able to fabricate arbitrary messages, e.g., fake DNS records and CSRs, but not break cryptographic primitives, e.g., forge signatures or the break symmetric encryption used in secure tunnels, such as TLS connections. The Tamarin model considers the possibility of keys being revealed to an adversary, e.g., due to compromised equipment. However, our threat model assumes that certain entities, such as parent domains of the victim, do not reveal any secret keys.

E.1.2 Domain Owner Requirements. Following our design, our model makes the following assumptions about the domain owner policies: we assume that the domain owner issues a policy for their domain as a CAA record which includes their caOwnerID, e.g., ACME account URI, or which specifies a secure validation method, and that CAs can securely look up this policy record, i.e., the record is covered by DNSSEC and a valid DNSSEC chain exists.

E.1.3 Secure Channel Assumptions. Following our threat model, we assume that (1) the domain owner and all parent domain owners are benign, that (2) all domain owners have authentic communication channels to their parent domain owners for updating their

DNS(SEC) records, and (3) that domain owners can create CA accounts and that CAs can authenticate their users through their account URI.

Secure local computations. The following protocol steps are performed locally at a benign domain owner or DNS nameserver and are thus assumed to be secure:

- Create TLS and DNSSEC keys
- Sign TLS transcripts
- Choose the desired validation method
- Create SIG, CAA, NSEC records

Secure DNSSEC setup. We assume that the victim domain owner is able to securely communicate with their parent domain, to request the desired DNSSEC records, and with their DNS nameservers to upload the required DNSSEC records:

- Benign domain owners can upload all DNS(SEC) records to the correct DNS nameserver
- Benign DNS nameservers only sign correct NSEC records and only sign DS records intended by the domain owner

Secure CA account creation. We assume that a domain owner can securely create a CA customer account and retrieve the correct values to perform validation. Note that by itself, this does not enable cryptographic domain validation, i.e., the domain owner must still prove domain ownership to the CA after securely creating an account:

- The domain owner can create a CA (e.g., ACME) account with a unique account URI
- The domain owner can retrieve the correct DNSSEC validation nonce from the CA
- The CA knows which account URI a certificate issue request belongs to

E.2 Protocol Modeling

We base our model as closely as possible to the relevant protocol specification documents. However, since DNS and DNSSEC are extremely complex protocols described in dozens of extensive standardization documents, it is beyond the scope of this formal analysis to completely include all protocol intricacies. Concretely, our model simplifies the following aspects of DNS and DNSSEC, while ensuring that we can still provide strong guarantees for the security properties of cryptographic DV.

(1) **Temporal ordering.** Instead of considering concrete time intervals, e.g., the ability of a CA to reuse a prior validation for 398 days [16], our model only captures the ordering of actions, e.g., whether a DNSSEC-signed CAA record was created before or after a certificate was issued. However, once an attack is discovered, it can easily be instantiated by the concrete time intervals to showcase the potential magnitude of the attack.
(2) **Protocol-specific modeling.** We change certain protocol-specific encodings to a Tamarin-compatible format and do not model auxiliary records and fields that are not used for certificate validation. For example, we do not model the salt and iteration count for NSEC3 records, as it is solely used for privacy purposes and has no impact on providing authenticated denial of existence [35]. Furthermore, we only consider the "online KSK" mode, where the KSK is stored together with the ZSK, i.e., entities do not reveal only a KSK or only a ZSK.
(3) **Revocation.** We do not explicitly model revoked DNSKEY records (RFC6781 [51]) and instead treat revoked keys as non-existent keys.

E.3 Prior and Future Work

Tamarin has been used in prior Web PKI research, such as ARPKI [6] and DTKI [77]. Compared to previous formal models, e.g., the RFC-based model of ACME protocol [7], our model focuses on the interaction between the relevant actors in cryptographic DV instead of low-level protocol details. Wrotniak et al. formally model issuance, revocation, and validation in the Web PKI [75]. Our model is focused on the aspect of domain control validation, and is thus orthogonal to their general PKI model, which could be instantiated using cryptographic DV.

As future work, our model could be extended with a more precise model of interactions between domain owners, DNS registrars, DNS registries, and DNS nameservers. Furthermore, our model could be extended to capture the interactions of the communication between domain owners and CAs during the certificate issuance protocol, e.g., by integrating it with the ACME security analysis by Bhargavan et al. [7].

F CA Accountability

In the main body of the paper, we show that our protocol can achieve strong security guarantees, such as preventing network attacks as shown in Table 3, under the assumption that all CAs follow the CA/Browser Forum guidelines to perform validation. In this section, we expand our threat model to include misbehaving CAs, e.g., CAs that do not properly perform domain validation or forge or misinterpret validation results, and elaborate on how our work has important synergy with Certificate Transparency to hold misbehaving CAs accountable.

F.1 Challenges in Achieving Accountability through CT

The problem of misbehaving CAs has already been observed over a decade ago with the notorious DigiNotar incident [45] and sparked the development of accountability mechanisms for CAs, i.e., certificate transparency (CT). Recall that CT ensures that all certificates issued by a CA are included in a publicly auditable append-only log structure, allowing independent auditors to detect fraudulently issued certificates and either revoke the certificate or even remove the issuing CA from the root store. One of the drawbacks of CT is that the detection of fraudulent certificates and the auditing procedure requires a significant amount of manual labor. Automating the detection of fraudulent certificates is challenging since there is no reliable source of historical data on which entity controls which domain. For example, relying on CAA records, which are used to govern certificate issuance, is problematic for an automated detection system, since a change in CAA records may lead to correctly issued certificates spuriously being flagged as fraudulent. This risk of false positive either makes the system unreliable or requires

Table 5: Attack prevention against benign CAs and detection against misbehaving CAs. The columns authorized and unauthorized CA refer to misbehaving, i.e., non-benign, CAs that are and are not listed as allowed issuers in the domain's CAA record.

Mechanism	Prevents attacks on benign CAs	Detects attacks by misbehaving CAs	
		Unauthorized CA	*Authorized CA*
No DNSSEC Enforcement	○	○	○
Cryptographic Domain Validation	●	○	○
Logging CAA DNSSEC Verification Result	●	●	○
Logging CAA + CSR DNSSEC Verification Result	●	●	●

manual vetting of the results, which defeats the purpose of having an automated system.

Similar limitations appear when using existing CT log entries to verify whether cryptographic DV was correctly performed by a CA. In particular, an independent auditor only observes the issued certificate and cannot deduce (1) if the issuing CA was authorized by an existing CAA record, (2) if an intended validation procedure (according to the CAA record) was used, and (3) if the validation was performed correctly.

F.2 Logging CAA DNSSEC Verification Result

The first issue of an unauthorized CA (as per the applicable CAA record) issuing a fraudulent certificate can be addressed as follows. Each CA must make all DNS(SEC) records that are relevant for checking the CAA record during domain validation available to external auditors (e.g., via an extension to CT logs). This allows an auditor, which could be an automated system, to fetch these records and verify either that there are no CAA records, i.e., a DNSSEC authenticated denial of existence proof, or that a CAA record for the issuing CA, including a valid DNSSEC chain, exists. We envision that this logging infrastructure could be implemented in two ways. The CA/Browser Forum could mandate that all CAs provide a public interface that returns these records, which can be queried by an auditor for a specific certificate identifier, similar to OCSP[2]. Alternatively, the IETF and the CA/Browser Forum could extend CT to include these records along with the certificate in the CT entry. The full design of this CT extension has several intricacies which are beyond the scope of this paper. For example, CT logs must prevent denial-of-service attacks by not allowing the inclusion of arbitrary (large) DNS records. Similarly, framing attacks, where an adversary uploads a certificate with insufficient CAA verification results to accuse the CA of misbehaving, must be prevented, e.g., by including a commitment from the CA to a given set of DNS records in the CT entry.

F.3 Adding Domain Owner CSR Commitments

The second and third issue, namely auditing whether the correct validation procedure was used by the CA and whether the validation was performed correctly, cannot be solved through logging with current domain validation methods. Instead, a domain validation must produce cryptographic proof of its correctness. For example, even if a CA logs a valid DNSSEC-protected DNS-01 record, an external auditor has no way of verifying that the issued certificate corresponds to the certificate requested by the domain owner in its certificate signing request (CSR). Instead, to detect misbehavior of CAs that are authorized for a domain, the domain owner can create a DNSSEC-protected DNS record for its domain which contains a binding to its CSR, or relevant parts of the CSR, such as the domains, issuer and public key. This binding could be a serialization of the relevant data, or the hash of the serialization, as long as the auditor can re-construct the hash input from the resulting certificate. Additionally, the domain owner may specify a start and end time during which the issuance of this certificate is allowed. This binding could either be added in a new DNS record or as a validation method field in an existing CAA record. This would work both for existing "issue" tags or our proposed "security" tags.

We provide an overview of the effectiveness of various validation protocols in preventing and detecting attacks in Table 5.

G A look at top domains with cryptographic DV elements

We found 74.8K domains who are early adopters of cryptographic DV elements. This includes many popular websites in the top 1M Tranco [56] site ranking (generated on 1 May 2024 [3]). We found official sites that host several popular OSes (Debian, GrapheneOS), the Bitcoin Core source code, the Slack messaging app, and the XMPP-based messaging app Jabber all have CAA tags properly configured for cryptographic DV. Some domains did have an issue tag for cryptographic DV but also contained additional issue tags that made them insecure (e.g., torproject.org). A selection of these high-impact domains is included in Table 6. The full list of domains using cryptographic DV can be downloaded here.

[2]Note that in contrast to OCSP, these responses may not need to be signed as they are independently cryptographically verifiable.

[3]Available at https://tranco-list.eu/list/7XN4X.

Table 6: Details of issuance policies for top ranked domains using cryptographic DV elements. Domains marked in red have additional CAA "issue" tags that permit insecure issuance. Cryptographic DV additionally requires that DNSSEC validation by CAs be standardized.

Domain name	Ranking	Domain policy contents			
		DNSSEC signed?	Account ID?	Method dns-01	Restricts insecure issuance
slack.com	181	✓	✓		✓
debian.org	458	✓	✓	✓	✓
slackb.com	2169	✓	✓		✓
slack-edge.com	3092	✓	✓		✓
torproject.org	5005	✓	✓		
slack-imgs.com	6881	✓	✓		✓
samba.org	16674	✓	✓		✓
grapheneos.org	35880	✓	✓		✓
netzone.ch	43752	✓	✓		
max.gov	47237	✓	✓		✓
nmugroup.com	52237	✓	✓		
csswg.org	78210	✓		✓	✓
grapheneos.network	87939	✓	✓		✓
slack-files.com	92189	✓	✓		✓
torproject.net	96610	✓	✓		✓
projectsegfau.lt	98290	✓		✓	✓
browserleaks.com	114243	✓		✓	✓
jabber.ru	151850	✓	✓	✓	✓
bitcoincore.org	217237	✓	✓		✓