Cryptographically-Secured Domain Validation

Henry Birge-Lee [*] *Princeton University*
Grace H. Cimaszewski [*] *Princeton University*
Cyrill Krähenbühl *Princeton University*
Liang Wang *Princeton University*
Aaron Gable *Let's Encrypt*
Prateek Mittal *Princeton University*

Abstract

Certificate Authorities (CAs) bootstrap HTTPS-based trust on the Internet by authenticating the identity of domain names via a process known as domain control validation (DV). Because of their reliance on unauthenticated web protocols (including DNS and HTTP), currently used DV protocols have been shown to be vulnerable to a range of network attacks. To remedy this situation, we propose a framework for cryptographic DV, which fundamentally mitigates network-layer attacks. Our framework offers secure DV protocol mechanisms that are coupled with a secure policy specified by the domain owner, to remove reliance on plaintext communication that renders DV vulnerable to network attacks. The design leverages existing pieces of the current Web PKI and DNS ecosystems, such as Certificate Authority Authorization (CAA) policies and secure DNS, to ensure backwards compatibility and ease of deployment. We showcase the feasibility of our framework through collaboration with a major CA, where parts of our design have been implemented in its production deployment and over 74 thousand domain names have gained certificates via cryptographic DV methods. We demonstrate the strong security properties of our design formally using the Tamarin verification tool and empirically via ethically-conducted real-world attacks. We further analyze the deployment potential of our framework via a longitudinal analysis of over 600 million domains and their DV processes. Overall, our cryptographic DV design provides domain owners the critical capability of declaratively securing their domain names against network attacks on certificate issuance.

1 Introduction

Certificates issued by the Web PKI are the fundamental trust mechanism underpinning HTTPS on the web today, which provides the important security properties of confidentiality, integrity and authenticity. Obtaining a certificate for a domain name involves proving control over the 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 true domain owner can perform, such as publishing a text file containing a nonce on the webserver or a DNS record. However, DV itself is vulnerable to attacks because it involves unauthenticated protocols, such as DNS and plaintext HTTP as shown in Fig. 1. This crucial flaw means that certificates' authenticity properties can be subverted by network attacks, and attackers can gain certificates for domain names they do not actually own [8, 11]. Such attacks are feasible through BGP hijacks or DNS hijacks, and have already been observed in the wild [7]. We are tasked with an important challenge: can we fundamentally mitigate the reliance on untrusted channels to bootstrap trust in today's Web PKI?

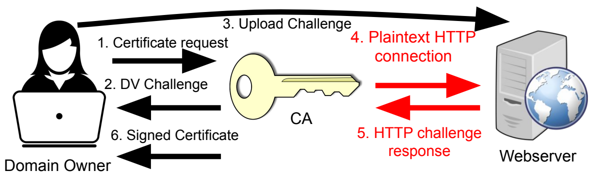


Figure 1: Domain Control Validation via HTTP.

The Web PKI security community has responded with a range of solutions which have been adopted to varying extents [9, 25, 38, 47]. However, none have fully resolved the fundamental security 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 [9, 14, 18]. Certificate Transparency (CT) provides detection of misissued certificates but does not prevent their issuance [47]. DANE and HPKP give domain owners the ability to more tightly restrict their domains' certificates, but suffer from serious availability prob-

[*]Both authors contributed equally to this work.

lems and never saw long-lived adoption [25, 38]. Finally, several proposals for enhancing PKI security, e.g., by leveraging certificates from multiple CAs, still require that CAs can securely perform domain validation [5, 16, 43, 60, 61, 66].

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, 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 mechanism that meets all of these qualities while providing cryptographically-secured DV for any opt-in domain. Our design consists of two main 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 must be followed by CAs while 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)[1]. This approach does not introduce any new infrastructure components, which not only makes it fully interoperable with current certificate issuance processes but also minimizes 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 domain validation methods then utilize cryptographic keys defined in these policies to mitigate vulnerabilities in current DV mechanisms that rely on insecure channels. One of the world's largest CAs has implemented parts of our framework in its production deployment, demonstrating our design's practicality.

[1]While DNSSEC validation is not currently supported by all CAs, this can potentially be mandated in the future by the CA/Browser Forum.

We rigorously estimate the effectiveness and feasibility of our design through a three-pronged analysis. First, 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. Second, 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. Third, we perform a longitudinal analysis of a dataset from a major CA's production deployment logging the issuance of certificates for over 600 million domain names to demonstrate the feasibility of domain owners using our design. We find that over 74K live domains already use secure DV methods implemented by this CA.

Our work closely builds upon RFC 8657, which introduces DNS CAA extensions (called ACME CAA) for domain owners to specify granular issuance policies to CAs [46]. Our issuance policies go beyond RFC 8657 by offering declarative security descriptors that hold for all domain validations, enabling domain owners to secure certificate issuance for all CAs, e.g., increasing resilience through automated CA failover. Our cryptographic DV leverages such issuance policies and proposes a complete system with robust security guarantees for the PKI ecosystem. Our validation methods have complete security guarantees proven through formal analysis and are motivated by concrete web/DNS hosting and certificate issuance behavior by real domains (as evidenced over several months of a CA's issuance logs).

Our work envisions a world in which neither rogue hackers nor powerful nation states can target the Web PKI and compromise confidentiality, integrity, and trust in our web communications. Our design tackles the central challenge of establishing trust on the Internet today and enables domain owners for the first time to *fundamentally protect themselves against network-layer attacks on certificate issuance.*

2 Limitations of Previous Proposals to Secure Certificate Issuance

In the HTTPS ecosystem, misissued certificates pose a critical security vulnerability. An adversary can use a misissued certificate to machine-in-the-middle a TLS connection and pose as a victim's server, compromising both confidentiality and integrity of communications. This attack has occurred in several real-world incidents [7, 42].

Several technologies have arisen to improve the security of digital certificates (full discussion in Section 6). However, the technologies that have seen widespread deployment have typically focused on attack detection (e.g., certificate transparency) or enhancing the difficulty of attacks (e.g., MPIC). These protocols are complementary to our proposed solution for securing the domain validation of benign CAs, i.e., CT addresses the concern of *misbehaving CAs*, while MPIC de-

Table 1: Design characteristics of proposed PKI improvements. HPKP and NOPE do not prevent networks attacks on the first connection while MPIC only prevents localized attacks.

Design	Prevents Attacks	No Client DNS/TLS Changes	No Client Software Changes	Allows Public Key Rotation	No Domain Webserver Changes	No Domain DNS Changes
Certificate Transparency	○	●	●	●	●	●
(DANE) Mode 0 CA Specification	○	○	○	●	●	○
HPKP	◐	○	○	○	○	●
NOPE	◐	●	○	●	○	○
MPIC	◐	●	●	●	●	●
(DANE) Modes 1 & 3 Specific TLS Certificate	●	○	○	○	●	○
(DANE) Mode 2 Trust Anchor Assertion	●	○	○	●	●	○
Cryptographic DV	●	●	●	●	●	○

fends all domains against *local network adversaries* without requiring domain owner changes. While several standardized proposals have aimed to prevent attacks (e.g., DANE), none have become best-practices due to deployment challenges discussed in this section. We compare our work with notable past techniques in Table 1.

Certificate Transparency: Focusing on detection not prevention. Certificate Transparency (CT) allows for the detection of misissued certificates via append-only logs operated by third parties. CAs are required to upload *precertificates* (which indicate a CA's intent to sign a certificate) to CT logs and then include a Signed Certificate Timestamp in the corresponding signed certificates. This provides cryptographic proof to browsers that a certificate was included in a trusted CT log. By providing visibility into issued certificates, the logs enable independent parties to identify fraudulent certificates. However, as a reactive measure, *CT cannot prevent the issuance of a malicious certificate.* Furthermore, as inclusion of a certificate into the logs may take up to 24 hours, CT does not enable real-time misissuance detection [2]. Domain owners must also proactively sign up for CT log monitoring and initiate revocation requests themselves. As an example, misissued certificates in real-world attacks such as the CelerBridge and Klayswap attacks were indeed recorded in CT logs, but were not detected until the fallout of the attacks (i.e., cryptocurrency theft) became apparent [7, 42]. Past certificates recorded in CT logs may offer an alternative way of securing domain validation by enabling a domain owner to prove ownership of previously issued certificates, but the CT monitoring and archival services needed for this design are often incomplete and the system fails open in cases where domain owners lose access to past certificates or domain ownership changes hands [10, 51]. Our work advocates for enhanced security measures to prevent the issuance of fraudulent certificates due to weaknesses in domain validation protocols.

Multi-Perspective Issuance Corroboration: Partial protection against misissuances. Multi-Perspective Issuance Corroboration (MPIC) is a promising technology that hardens the certificate issuance process against equally-specific BGP attacks [59]. MPIC has been standardized by the CA/Browser Forum [14]. MPIC involves repeating domain control validation and CAA lookups at multiple remote perspectives spread across the Internet. MPIC helps mitigate equally-specific BGP attacks which tend to be localized to specific topological regions of the Internet (depending on the adversary's location) and thus are limited in their ability to affect all of a CA's validation perspectives. While this is a substantial improvement over the status quo, *MPIC only offers heuristic security:* subprefix BGP attacks and some equally-specific BGP attacks that affect all vantage points can still defeat MPIC. Design enhancements such as randomized vantage point selection and choice of quorum policy can improve MPIC's effectiveness but still fail to provide provable security properties [26]. Furthermore, MPIC does not protect against DNS cache poisoning [22] or other network attacks that can potentially affect all of a CA's validation perspectives.

DNSSEC-based Approaches: Requiring client and server changes. DNS-based Authentication of Named Entities (DANE) is an IETF-standardized proposal to include PKI information in DNSSEC records. 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 see widespread adoption largely due to the challenge of browsers reliably authenticating DNSSEC records [41]. Because DANE records can introduce new trusted certificates for websites, these records *must* be authenticated by DNSSEC and clients need to obtain full DNSSEC chain information to verify the records' authenticity. However, most DNS resolvers do not relay DNSSEC information to clients. Even if a DNS resolver does validate DNSSEC, it often performs the validation locally and simply returns the record type requested in the question with the Authenticated Data (AD) bit set, without the signature and key records needed for the client to independently authenticate the response. This highlights an important point in the web ecosystem: some protocols (particularly DNS) are calcified by decentralized legacy infrastructure and middle boxes [40]. Thus, changes that require clients adopting new DNS behavior can be problematic. NOPE [23] is a recent proposal for including small zero-knowledge proofs into Web PKI certificates that attest that a DNSSEC-protected record containing the certificate's public key existed at the time of certificate issuance. NOPE addresses the challenge in DANE of providing DNS(SEC) records to the client. However, NOPE is susceptible to downgrade attacks on initial (non-pinned) TLS connections, and compared to DANE, additionally requires software changes at the webserver.

HPKP: Hinders key agility. HTTP Public Key Pinning (HPKP) improves the security of the PKI by implementing a Trust-On-First-Use policy. HPKP is a HTTP header that implies a specific leaf certificate and/or additional certificates (e.g., intermediate certificates) in the certificate chain should be observed for TLS connections to that website. Once this trust on first use is established, even if an alternate malicious certificate is signed for the website, clients that saw the previous HPKP header will reject the new certificate. Several flaws with HPKP led to its deprecation [34]. First, it does not prevent attacks against first-time connecting clients that lack pinning information. Secondly, certificate pinning can be exploited for a denial of service attack. If an adversary hijacks a website and pins a malicious certificate that the victim does not control [33], even after the victim regains control of its website, clients may be unable to access it until the pin expires. An important lesson for new defenses is to enable *public key rotation*. If control of a website changes or new infrastructure is configured, past keys should not be strictly required to obtain a new certificate.

PKI redesign proposals. Proposals for more fundamental PKI (re-)designs, such as AKI [43], ARPKI [5], PoliCert [61], Cothority [60], DTKI [66], and F-PKI [16], provide strong security guarantees even under the assumption that multiple CAs and/or transparency log servers are compromised. These proposals typically require additional infrastructure to be deployed and changes at the clients, e.g., web browsers. We solve a complementary problem of how benign CAs can cryptographically secure domain validation, which all of these proposals ultimately rely on.

3 Design

In this section, we present our approach for cryptographically securing domain validation. Our design is informed by the realization that while we want to prevent attacks on certificate issuance, we are not blessed with the liberty of a clean-slate approach. Thus, we developed our design to not only secure domain validation, but to be interoperable with existing systems and to be incrementally-deployable. Overall, our design involves 1) domain owners uploading a policy in DNS using CAA records, 2) DNS operators using `authenticated DNS` (e.g., DNSSEC or recursive-to-authoritative DoT/DoH) to protect these policies, 3) CAs performing domain validation with encrypted protocols. This design also 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. 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. 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.

To secure domain validation, 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 perform it properly. Our threat model assumes that the security properties of DNSSEC or an equivalent secure retrieval mechanism are not compromised (e.g., via attacks that exploit improper DNSSEC configurations, insecure DNSSEC algorithms, or compromised parent domains that could take over control of the victim's domain). We also assume that DNSSEC-validating resolvers will always fail closed (i.e., return SERVFAIL) per the RFC specifications [37] 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. Our goal is prevent an adversary from obtaining fraudulent certificates via attacks on the domain validation procedures under our threat model. To achieve this goal, we develop a framework to embed authenticity, authorization, and transparency in DV procedures. A CA should have ways to verify the integrity of important resources relevant to DV to ensure they have not been tampered with (authenticity) and ways to limit processing of DV requests from anyone other than the authorized domain owner (authorization). Furthermore, the legitimate domain owner should have ways to express her DV preferences, such as authorized CAs and preferred secure validation methods, in a public way to prevent downgrade attacks that trick a CA into using insecure validation methods (transparency).

Design philosophy. As discussed in Section 2, existing techniques like DANE face challenges in deployment (and interoperability) 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, (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, and (3) centralization of maintenance at few parties could reduce human errors. 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. Security improvements to the PKI should be interoperable with existing technologies. Communication must be able to proceed using established legacy protocols even if one endpoint does not support the new protocol, and should not be disrupted by existing middleboxes or network infrastructures. For example, the client should be able to engage in secure communication even when using legacy DNS client software and resolvers (a challenge which has hindered DANE). Existing webserver software and CDNs should be able to serve websites that opt in to our new protocol without requiring any changes to the web hosting infrastructure. Finally, our design prioritizes interoperability while remaining robust against downgrade attacks.

Incremental deployability. We aim for our design to 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.With this change in place, our design prevents downgrade attacks against non-participating CAs.

Key agility and availability. The current PKI offers flexible, on-demand key changes and supports multiple keys for the same destination. Many PKI applications operate as fail-open regarding revocation, allowing TLS connections to proceed even without knowledge of the certificate's revocation state. This resilience is a chief strength of the PKI, allowing existing client-server TLS connections to continue without interruption even if PKI-related infrastructure goes down. A lesson from HPKP's deprecation is that any new improvement to PKI should preserve key agility and availability.

3.3 Cryptographic Domain Validation

Conventional DV protocols have two security flaws: (1) lack of resource (DNS records, domains, etc.) authentication and authorization during certificate request processing and (2) lack of ways for domain owners to explicitly communicate their intentions to use specific validation methods (e.g., our cryptographic domain validation methods) to specific CAs. These allow the adversary to forge certificate requests for victim domains or fool the CA to use insecure validation methods which can be thwarted by an attack.

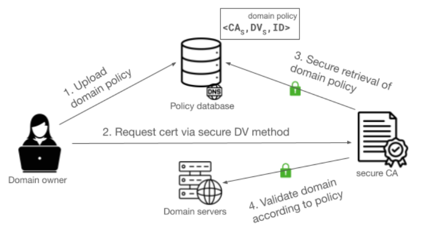
3.3.1 Overview

To enhance the security of DV, we introduce *secure validation methods* to provide authentication when validating the domain owner's control over resources, *domain ownerID (key)* to facilitate certificate request authorization, and secure discovery of *domain owner policies* to enable the domain owner to explicitly express her validation preferences to CAs.

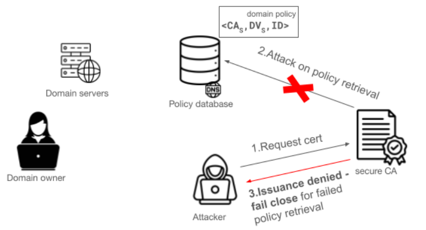
In cryptographic DV (Fig. 2), a participating domain owner specifies a policy that (1) constrains the set of DV methods that a CA is allowed to use such that the allowed methods provide authenticity guarantees, and (2) constrains the authorized entities that are allowed to request certificates from a CA. When the legitimate domain owner requests a certificate, the CA first securely retrieves the domain policy, and uses the owner-specified policy for DV and certificate issuance. The CA uses the owner-specified secure methods to validate the domain owner's legitimate control over domain resources. The CA can further validate that the certificate request is from an authorized entity by checking credentials specified in the domain policy (domain *ownerID*).

In our design, we take deployability into account and ensure that each mechanism can independently provide partial security improvements to DV. When combined, these mechanisms not only guarantee cryptographic DV, but also enhance the security of the entire lifecycle of certificate issuance. We also bootstrap our methods with established technologies. For example, we leverage DNS CAA records [30] to express domain owner policies (Section 3.4).

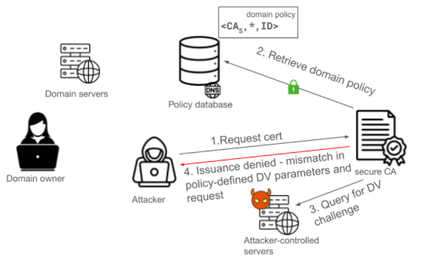
Challenges. While certain aspects of our proposal may sound straightforward in concept (e.g., using secure validation meth-


domain policy
<CA_S, DV_S, ID>
1. Upload domain policy
Policy database
3. Secure retrieval of domain policy
2. Request cert via secure DV method
Domain owner
secure CA
Domain servers
4. Validate domain according to policy

(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.


domain policy
<CA_S, *, ID>
Domain servers
Policy database
2. Retrieve domain policy
Domain owner
1.Request cert
4. Issuance denied - mismatch in policy-defined DV parameters and request
secure CA
Attacker
3. Query for DV challenge
Attacker-controlled servers

(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 2: The secure DV framework (2a) protects against attacks on secure policy lookup (2b), and on DV challenge (2c).

ods for validation), its practical implementation is far more complex due to the many challenges that must be addressed: which technologies are best suited to instantiate these building blocks and how they can effectively be used to perform DV securely, what the potential deployment hurdles are, and how to prevent opt-in CAs from downgrade attacks in partial deployment. The intricacies of the PKI ecosystem introduce layers of complexity in design, where a seemingly minor oversight can lead to a significant security failure. Overcoming these challenges requires a systematic investigation of the candidate technologies, and robust design.

3.3.2 Secure validation methods and *ownerID*

We define a *secure validation method* as one that performs the DV challenge-response protocol over a secure (authentic but not necessarily confidential) channel. Existing validation methods typically use insecure (plaintext) channels allowing adversaries to compromise the security of DV. We therefore propose that participating domain owners only use secure validation methods in cryptographic DV. This immediately raises a "chicken-or-egg" dilemma: there is a circular dependency between the PKI's role in bootstrapping secure channels and its own reliance on secure channels for domain validation. To break this circular dependency, a key design insight is that CAs can bootstrap secure validation methods using trust anchors such as `authenticated DNS`. For certificate renewals, when an existing (valid) TLS certificate is available, this dependency can be broken via TLS.

We also introduce *ownerID* in cryptographic DV to enable the CA to verify whether a certificate request is coming from an authorized user. This can be used to provide cryptographically-provable security even if a CA used plaintext DV methods. The *ownerIDs* can be either CA-specific unique identifiers of user accounts (e.g., ACME accounturi [46]), or public keys bound to the owner's identity. The CA can choose to validate the requester's control of the corresponding credential or private key during DV. By including her *ownerID* in both the certificate requests and resources, the domain owner can create a strong binding between her identity, certificate requests, and the controlled resources. Only the legitimate domain owner can demonstrate the control of appropriate credentials or private keys to the CA, which prevents impersonation. The concept of *ownerID* creates another circular dependency: a CA must know a domain's *ownerID* to verify it, but can't learn it without prior knowledge. We break the dependency by publishing *ownerID* in domain owner policies secured via authenticated DNS (Section 3.3.3).

Secure validation methods allow a CA to verify a domain owners' real-time control of web/DNS services while *ownerID* methods allow a CA to verify if the certificate request is authorized. While both approaches can be used for cryptographic DV, they are trade-offs. Secure validation methods which verify real-time control over web/DNS resources are compliant with CA/Browser Forum. On the other hand, *ownerID*-based methods cryptographically prove owner identity via static credentials but are not CA/Browser Forum compliant. Thus, these methods need to be used with an online challenge-response protocol to be CA/Browser Forum compliant. For defense in depth, we advocate for a combination of secure validation methods and *ownerID*.

CA/Browser-Forum-compliant DV methods.

(1) secure-dns-record-change: The CA performs a DNS-based challenge that requires the requester to show the control of a DNSSEC-signed DNS record, or resolve all DNS records related to the challenge over an authenticated channel, e.g., recursive-to-authoritative DoH/DoT.

(2) http-validation-over-tls: The CA performs an HTTP-based challenge with the target domain over a TLS connection (on port 443) using a trusted certificate. This can be accomplished either by the CA directly connecting to an HTTPS URL at the domain being validated or arriving at an HTTPS URL at the domain being validated after following a redirect. This method is compatible with the ACME standard even

thought the standard mandates the "http-01" method involve a plaintext connection to port 80. Domains that redirect HTTP traffic to HTTPS can comply with both ACME "http-01" and http-validation-over-tls.

DV methods used in conjunction with CA/Browser-Forum-compliant methods.

(3) known-account-specifier: The CA validates that the requester has access to a subscriber account that is linked to the target domain via a cryptographically-authenticated DNS record (e.g., an ACME account URI stored as a *caOwnerID* in the domain owner policy, see Section 3.3.3). Once the CA validates the requester indeed has control of this account, it must use a CA/Browser Forum compliant domain control validation method to proceed with issuance.

(4) private-key-control: The CA validates the requester's control of a private key that corresponds to a public key specified as a *globalOwnerID* in the corresponding domain owner policy the CA retrieves. After validating the *globalOwnerID*, the CA must use a CA/Browser-Forum-compliant domain control validation method to proceed with issuance.

3.3.3 Secure domain owner policy

In the 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 (if the *ownerID* is not used); or, the adversary could request certificates from a CA that does not support cryptographic DV. To prevent such downgrade attacks, we propose a *domain owner policy* to allow the domain owner to express their preferences for domain validation, such as the allowed secure validation methods, preferred CAs, and/or *ownerIDs*.

Expressing domain owner policies. A domain owner policy is represented as a set of 4-tuples that have the form:
$<$caName, validationMethod, caOwnerID, globalOwnerID$>$

caName specifies a CA that is authorized by the domain owner to issue certificates for their domains. *validationMethod* specifies the validation method the domain owner wants the CA(s) to use. The *caOwnerID* specifies the identity of the domain owner in the context of a specific CA, while the *globalOwnerID* specifies a global identity of the domain owner, usable across different CAs. The *globalOwnerID* must be tied to a public key, for which the domain owner controls the corresponding private key. The policy can either specify just one of *validationMethod*, *caOwnerID*, *globalOwnerID* or any possible combination of these options. Furthermore, if a single domain policy comprises multiple policy 4-tuples, a CA that is compliant with any 4-tuple may issue a certificate. The policy provides flexibility to allow the domain owner to select the CAs and validation methods to use for a target domain, and block issuance by any non-participating CA that does not support our approach.

Accessing domain owner policies. Rather than relying on the preferences specified in the certificate request (which could be from an adversary), the CA must be able to retrieve the legitimate domain owner policy directly during validation. A lightweight approach to make the policies accessible to CAs is to store the policies as DNS CAA records on the authoritative DNS servers of the domain. The CA/Browser Forum mandates that CAs must perform and adhere to the CAA checking requirement during domain validation, which means using CAA records to express domain policies would not require any additional communication ([13], §3.2.2.8). A CAA record consists of three major parts: (1) Flag: an integer between 0 and 255, with the property that any value over 128 (i.e., has the high-order bit set) indicates that the CAA record is *critical*, and any CA that does not understand the parameters must not issue a certificate. (2) Tag: a string that defines the action taken by the CA, and the CAA standard [30] allows CAs to define their own tags. (3) Value: the specific policies associated with the tag. CAA records can be flexibly extended with customized properties. Hence, we focus on using CAA records as a platform for expressing broader domain owner policies related to domain validation.

Secure discovery of domain owner policies. 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. In the case that CAAs are used for storing policies, we propose the 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 or performed over a secure channel with the appropriate authoritative server(s).

Properties of `authenticated DNS lookups`. An `authenticated DNS lookup` mechanism must satisfy the following properties to protect domain owner policies. First, data retrieved from an `authenticated DNS lookup` must be protected against tampering by network-level adversaries. Second, an `authenticated DNS lookup` must not be downgraded to an unauthenticated DNS lookup (i.e., failure to authenticate the DNS lookup is considered 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 option to help achieve these properties. We require CAs to perform DNSSEC checks to either validate the authenticity of retrieved policy (DNSSEC-signed CAA records) or to validate a proof that no policies exist for this domain (DNSSEC-protected NSEC(3) records [28]) in order to proceed with issuance. If DNSSEC validation fails, we require the CA to fail-close and deny issuance.

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 [57] (or out-of-band channels). See Appendix A.1 for more details.

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, 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 5). Using secure channels for authenticated DNS can potentially protect many more domains with domain-owner participation. 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%+) [18]. Actually deploying such channels in a manner that satisfies the properties of `authenticated DNS lookups` requires some changes discussed in Appendix A.

3.4 Expressing Domain Owner Policies via Existing CAA Records

Our proposed domain owner policies can be expressed using RFC-compliant CAA records. An issue tag with a CA name as the value restricts certificate issuance to that specified CA. However, standard CAAs cannot stipulate validation methods and *ownerID*. CAA extensions, described in RFC8657 [46], allow the domain owner to define ACME account URI and allowed validation method in CAA records. This extension format can be used to express policies that contain *caOwnerID* and *validationMethod* for cryptographic DV.

While RFC8657 discusses the use of ACME Account URI (a form of *caOwnerID*) and DNS-based validation methods in conjunction with DNSSEC to protect domain validation from MitM adversaries [46], our work goes beyond this to provide a comprehensive framework to deploy cryptographic DV. We develop additional methods for performing cryptographic DV (http-over-tls and private-key-control) that cannot be expressed in RFC8657 syntax, and consider general DNS security beyond DNSSEC using the abstraction of `authenticated DNS lookups`. Finally, an integral part of our design is declarative security (discussed in Section 3.5) for cryptographic DV which RFC8657 records cannot achieve.

3.5 New CAA Tags for Cryptographic DV

While existing CAA tags can be used to implement our design, they lack *declarative security*. They rely on chaining security properties from various technologies *without allowing domain owners to explicitly declare the desired property of cryptographic DV*. This approach requires blocking all insecure validation methods rather than directly communicating the intent of using secure validation methods to CAs.

The lack of declarative security has several security implications. First, future changes in the DV ecosystem could weaken security. Consider the case where a CA suddenly switches from `authenticated DNS lookups` to plain-text DNS in DNS-based validations. A domain relying on DNS validation to provide cryptographic security could inadvertently be compromised by this change. Declarative security ensures the domain's intent for validation is communicated to the CA, requiring CA policy changes to align with the domain owner's requirements.

Furthermore, without declarative security, if a domain administrator misconfigures any of the building blocks that are being chained together for overall security, it could have a devastating effect because CAs will fail open. For example, the domain administrator has the burden of (1) correctly configuring the CAA record (including CAA extensions) and (2) ensuring all subdomains used in DV support `authenticated DNS lookups` such that the end result provides cryptographic assurance for domain validation. Achieving security without declarative policies 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 [56]. The use of existing CAA tags that lack declarative security causes misconfiguration issues to fail open because the domain owner's intent is never communicated to the CA. With declarative security, CAs fail close on configuration issues preventing many vulnerabilities (e.g., Appendix B).

Finally, using existing CAA records for secure validation requires listing all preferred CAs and manually updating DNS for alternatives if needed. While this limitation is accepted in today's PKI, a resilient future PKI needs automated CA failover, which current practices do not support.

Declarative security using a new secure CAA tag. To provide declarative security, we propose a new CAA tag that enables domain owners to specify desired security properties for domain validation instead of listing specific CAs. While CAs can optionally be listed, it is not required. This allows certificate management tools to select any participating CA that supports cryptographic DV, while keeping CAA records static (tolerating churn in participating CAs).

Specifically, we introduce a new tag security (Cryptographic DV) in CAA records. Table 5 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. This tag enforces cryptographic DV across all CAs without requiring domain owners to enumerate specific CAs or methods, and is seamlessly interoperable with current CAA operation in the CA/B Forum Guidelines, enabling partial deployment. Furthermore, the properties set with this tag are not CA-specific and can define validation parameters for **ALL** compliant CAs. This record represents a shift in perspective for CAA records, moving away from listing acceptable CAs to specifying the desired security properties for any CA.

We require the critical bit to be set in security CAA records. The semantics of the CAA critical bit enable security against downgrade attacks in the setting of a partial deployment scenario, as non-participating CAs that do not understand the CAA records must reject issuance. We also require the parent domain to block wildcard issuance to ensure subdomains' CAA records will be checked by the CA during DV. This tag provides stronger security properties than alternatives such as RFC 8657 extensions, namely declarative security properties across all CAs and support for non-ACME CAs (see Table 4).

Policies supported by our new CAA tag. The security CAA tag supports all protocols introduced in Section 3.3.2 for performing validation in a manner that is robust against network-level attackers. We can apply the domain owner policy format discussed in Section 3.3.3:
$<\text{caName}, \text{validationMethod}, \text{caOwnerID}, \text{globalOwnerID}>$

For reference, a regular CAA issue tag for CA X would look as follows: $\{<X, m, *, *> \mid m \in \text{VM}_{\text{CAB}}\}$, where VM_{CAB} are all CAB-approved validation methods.

An empty security CAA tag (i.e., no value given) encodes to the following policy:

$$\{<\text{ca}, m, *, *> \mid \text{ca} \in \text{CA}_{\text{OPTIN}}, m \in \text{VM}_{\text{SEC}}\}$$
$$\cup \{<\text{ca}, *, \text{ID}_{\text{ca}}, *> \mid \text{ca} \in \text{CA}_{\text{OPTIN}}\}$$
$$\cup \{<\text{ca}, *, *, \text{ID}_{\text{global}}> \mid \text{ca} \in \text{CA}_{\text{OPTIN}}\}$$

where CA_{OPTIN} are opt-in CAs that implement cryptographic DV and VM_{SEC} are secure validation methods.

Additionally, the security CAA tag can specify cryptographic DV methods. For example the tag "security methods(secure-dns-record-change)" encodes the domain owner policy $\{<\text{ca}, m, *, *> \mid \text{ca} \in \text{CA}_{\text{OPTIN}}, m \in \text{VM}_{\text{SEC-DNS}}\}$, where $\text{VM}_{\text{SEC-DNS}}$ refers to performing the ACME dns-01 validation method or the CA/Browser Forum DNS change validation method *over authenticated DNS*.

3.6 Achieved Properties

Security. Our proposal advances existing approaches to securing domain validation by allowing domain owners to explicitly define the security properties for their domains' certificate issuance. 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 enhances security for any participating domain owner, regardless of CA adoption. `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 2.

Table 2: 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

Key agility and availability. Our design ensures that legitimate domain owners can obtain and update 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 to update ownerID 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.

3.7 Implementation and Deployment

We present the experience of a major anonymous CA that implements *cryptographic DV* via the mechanisms in Sections 3.3.3 and 3.4. Specifically, the CA (1) implements *authenticated DNS lookups*; (2) implements recognition of *validationMethod* and *caOwnerID* domain owner policies within CAA records, and (3) implements the "known-account-specifier" and "secure-dns-record-change" validation protocols as described below.

Deploying authenticated DNS lookups. The CA validates DNSSEC for all DNS queries to signed zones, both for CAA retrieval and during domain control validation. As a publicly trusted CA, they are required by the CA/Browser Forum to operate their own recursive resolvers. In the 9+ years since the deployment of Unbound, DNSSEC validation has not caused any availability or correctness issues for the CA.

Implementing authenticated domain owner policies. The CA implements support for RFC 8657 [46], 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 a *caOwnerID* policy. Similarly, the "validationMethods" parameter allows the CAA record to specify one or more ACME validation methods (such as "dns-01") 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 offers domain operators the ability to opt in to cryptographic DV. To opt in, the domain operator (1) enables DNSSEC and (2) sets a CAA record with the accounturi parameter specifying their unique account identifier, the validationmethods parameter specifying the "dns-01" validation method, or both. 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 cryptographic DV via these mechanisms (Section 5).

Future work estimates. The CA estimates the effort of implementing the `security` CAA tag to be similar to the 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 was set in the response) and passing that boolean into the CAA-checking routine requires work comparable to implementing the validationmethods parameter.

4 Security Analysis

We verify the security guarantees of cryptographic DV through a combination of formal analysis and ethically conducted attacks. We develop a formal model using the Tamarin prover [52] 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 the usefulness of a concrete instantiation of cryptographic DV features to protect a CA from fraudulent certificate issuances due to on-path or off-path attacks.

4.1 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. Alternatively, the authenticity of domain owner policies could be protected via secure channels to DNS nameservers. We focus on DNSSEC since it is fully specified by mature standards, allowing us to derive a concrete model, while several aspects of the secure channel approach, e.g., downgrade attack prevention mechanisms, are not yet standardized, and design changes may require adjustments to the formal model. Nonetheless, we expect that given the additional assumptions in Appendix A, the secure channel approach can protect domain policies and prevent downgrade attacks.

Security Properties. We classify CAs based on two criteria. A "secure" CA properly validates DNSSEC-protected CAA entries, or their absence, and rejects issuance in case of a failed validation, while an "insecure" CA may omit validation steps or ignore validation results. Based on our empirical analysis shown in Table 6, at least three high volume CAs, namely Let's Encrypt, Google Trust Services, and Certainly, are already secure, and standardization efforts to mandate DNSSEC validation of CAA records for all CAs are ongoing. 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. In the security properties below, unless explicitly stated otherwise, the legitimate domain owner protects its domain through a DNSSEC-signed CAA record that specifies a caOwnerID

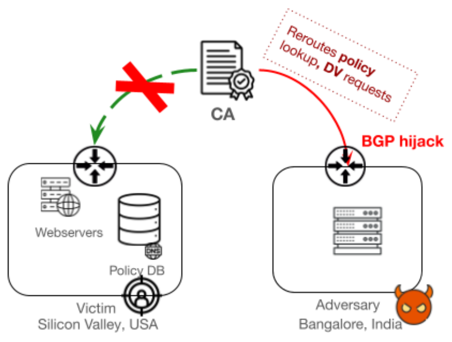


Figure 3: Setup for ethical attacks performed.

or a secure validation method, or both. Through our formal Tamarin model, described in Appendix C, we show that our proposed system achieves the following security properties:

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

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 domain owners that specify a caOwnerID or a secure validation method in their CAA records, and *set the critical flag*, are rendered ineffective.

4.2 Empirical Security Analysis

Next, we demonstrate the security of our approach by conducting concrete network attacks, enumerated in Table 2. 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 involves a victim domain name, hosted by a web server and DNS nameserver in a Silicon Valley cloud datacenter as shown in Fig. 3. We perform our experiments with two regimes of attackers: (1) an *on-path* adversary, modeled as a reverse proxy server situated on the path between the CA and the victim's web and DNS servers; (2) an *off-path* adversary with BGP-announcement capability hosted in a Bengaluru, India cloud datacenter, which performs separate BGP sub-prefix hijacks of the prefixes hosting the victim web and DNS servers. We perform these attacks against a "secure" opt-in CA that validates DNSSEC for CAA records and implements cryptographic DV extensions, and an "insecure" CA which does not validate DNSSEC for CAA policy lookup or DV traffic. We stress that the attacker's location is largely irrelevant for the takeaways of these attacks.

Table 3: 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

For the off-path case, sub-prefix BGP announcements have global scope; for on-path, any attacker on the path between the victim's infrastructure and the CA can perform attacks with similar effect. The tests used a customized BIND 9 DNS server and the Python library scapy to manipulate responses for domain policy lookups and DNS-based DV challenges.

Results. Table 3 describes the outcomes of a suite of network attacks against the 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 *caOwnerID* 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-DNSSSEC-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 CAA lookups aim to 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.

5 Feasibility Evaluation

While the security benefits of cryptographic DV apply fully to any opt-in domain irrespective of global adoption rate among domains, an open question is how many domains might smoothly choose to use these extensions. To this end, we evaluate customer domains in validation logs obtained from a research partnership with a major public CA. Our analysis points to significant prevalence of CAA record usage and DNS-based validation methods as well as stable usage

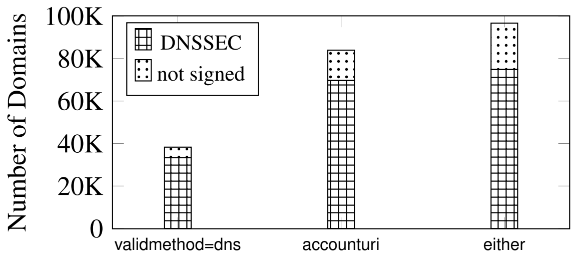


Figure 4: Usage of secure issuance policy elements in CAA.

of ACME account IDs, suggesting that domain owners are already familiar with most of the mechanics to implement cryptographic DV policies. We also observe that a small but significant number of "early adopter" domains already use secure issuance policies in practice.

CA Log Dataset. Thanks to our research collaboration with an anonymous CA, we receive daily logs of domain validation and certificate issuance events distributed across two datacenter locations hosted in the western U.S. The logs contain information on certificate requests not visible in public datasets such as CT logs, including validation type, logged responses of CAA record and DV challenge retrievals, and request information (account URI, etc.). We focus on results over 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* which requested certificates.

Takeaways. Our analysis suggests the following takeaways:

(1) A small core of early adopter domains already use the CA's cryptographic DV deployment: As shown in Fig. 4, a small fraction of domains are already using a cryptographic DV method in a DNSSEC-signed CAA record. 74.8K domains specify a DNS-based DV method or account ID in a DNSSEC-signed CAA record, which enforces that DV be performed with full cryptographic assurance, from resolution of the domain name itself up to retrieving and verifying the DV challenge contents. 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 high-impact domains using cryptographic DV is included in Appendix D. Of note is domain owners' preference for using account ID credentials: over 93% (69.7K) of domains using cryptographic DV specified account credentials in their issuance policies. Domain owners also tend to combine specifying a DV method *along with* account ID: nearly 91% of domains specifying secure DNS DV methods also specified an account ID. Finally, we highlight the high rate of DNSSEC adoption among these early adopters: over 77.4% of the 96.7K domains that specified a cryptographic-DV policy are DNSSEC signed, suggesting a high awareness of DNS security practices among these domain owners.

(2) There is significant adoption of the two main protocol prerequisites, CAA and DNSSEC. CAA and DNSSEC records are the foundation of our protocol. Though 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).

(3) Domain validation practices suggest that domain owners can 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 [4] and TLS-ALPN-01 [58] 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 customers 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 *caOwnerID* extension 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**, with 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.

Combined, these properties suggest that implementation of our extensions is feasible for CAs, and our mechanisms have already been used by a considerable number of domain owners. Besides, there exists a group of domain owners who would be incentivized to adopt our approach.

6 Related Work

In this section, we discuss related work beyond technologies already covered in Section 2. First, we cover additional defenses that mitigate the threat of network (hijacking) attacks. Second, we discuss broader techniques to remedy issues in the Web PKI beyond securing domain validation.

Attacks and defenses on domain validation. There is an extensive body of work which showcases how attackers can fraudulently obtain valid TLS certificates through weaknesses in BGP via prefix hijacks and interceptions [8,9,12,18,22,27], and in DNS(SEC) via DNS cache poisoning and insecure cryptographic agility [11,21,32].

The Resource PKI (RPKI) and Route origin Validation (ROV) offer limited protection against BGP hijack attacks by authenticating the binding between Autonomous Systems and prefixes they control [29, 35, 36, 53, 54, 65]. An alternative approach to defeat network adversaries is to probe a network resource from multiple perspectives [1,63], which is used in Multi-Perspective-Issuance Corroboration (MPIC) where CAs perform domain validation from multiple vantage points [9,11]. BGPsec [50] and SCION [17] mitigate hijack attacks by cryptographically authenticating both prefix origin as well as every AS hop in the routing announcements. However, BGPsec has so far not been deployed [45], and SCION deployment must further expand to become widely available. Even with a full BGPsec or SCION deployment, domain validation remains vulnerable to *on-path adversaries*. Our proposed cryptographic DV goes beyond this threat model and offers protection against *all network adversaries*.

Security beyond domain validation. An important body of work has focused on designing systems that address conceptual weaknesses of the Web PKI, such as lack of accountability and risk of compromised entities, which is essential for ensuring the PKI ecosystem's overall security. The threat model of many systems described in this section is 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.

In addition to DANE and HPKP discussed in Section 2, PoliCert [61], DTKI [66], and F-PKI [16] offer alternative proposals for domain policies, allowing domain owners to restrict the validity of their TLS certificates. The main difference between these policies and the policies introduced in our work is that these policies are aimed at client devices, e.g., an end-user using a web browser, and not the CAs. Similar to our approach, RFC 8657 [46] leverages existing CAA records to specify issuance policies to CAs. As discussed in Section 3.5, one limitation is that these CAA `issue` tags provide imperative security, i.e., requiring domain owners to specify eligible CAs as well as desired domain validation methods. In contrast, our cryptographic DV CAA tag provides declarative security. Our design also supports additional methods for performing cryptographic DV (http-over-tls and private-key-control) that cannot be expressed using the RFC8657 syntax.

AKI [43] and ARPKI [5] 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 [61] and Cothority [60] employ multi-signature schemes to issue and protect the domain policies and TLS certificates, respectively.

7 Conclusion

We advocate a vision for securing the central building block of the Web PKI, the domain validation protocol, via cryptographic guarantees. Our work tackles a long-standing bootstrapping problem in the HTTPS protocol: while HTTPS is designed to protect against network adversaries, the Web PKI introduces vulnerabilities in how trust parameters are established. Our work fundamentally addresses the vulnerabilities in the existing domain validation procedures by changing certificate issuance to use authenticated channels, thereby making HTTPS truly resilient against network adversaries.

Our approach can be easily deployed: it requires no changes to client-side software and provide security benefits with a single participating CA, as demonstrated via usage of our protocol at a major CA. Our only required global CA change, the usage of secure DNS lookup mechanisms, has already begun standardization in the CA/Browser Forum. Finally, we also see our work as a first step towards true CA accountability, as authenticated domain control validation can ultimately be recorded in CT logs and used as a proof of a CA's permission to issue a certificate.

8 Ethics considerations

Our attacks are conducted ethically via the following principles. The real-world network attacks discussed in 4.2 caused no disruption to real-world users or services. The experiments exclusively employed domain names, web servers, and DNS nameservers that are all under our control. All BGP announcements made impacted routing only for prefixes owned by us. Finally, we do not affect any real users in our tests, as all test prefixes and domains hosted no real-world services. As Web PKI-issued certificates are posted in CT logs, the identity of customer domains from the CA logs used in Section 5 does not pose privacy concerns to domain owners. DNSSEC usage and contents of CAA records of these domains are also not confidential, as domains' DNS records can be publicly looked up. We do not share any personally identifying details of domain owners from the logs, such as email addresses, IP addresses used for domain validation, or account IDs.

References

[1] Mansoor Alicherry and Angelos D. Keromytis. DoubleCheck: Multi-path verification against man-in-the-middle attacks. In *Proceedings of the IEEE Symposium on Computers and Communications (ISCC)*, July 2009.

[2] Apple. Apple's certificate transparency log program. https://support.apple.com/en-us/103703, November 2023.

[3] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. Resource Records for the DNS Security Extensions. RFC 4034, IETF, March 2005.

[4] R. Barnes, J. Hoffman-Andrews, D. McCarney, and J. Kasten. Automatic Certificate Management Environment (ACME). RFC 8555, IETF, March 2019.

[5] David Basin, Cas Cremers, Tiffany Hyun-Jin Kim, Adrian Perrig, Ralf Sasse, and Pawel Szalachowski. ARPKI: Attack resilient public-key infrastructure. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, November 2014.

[6] Karthikeyan Bhargavan, Abhishek Bichhawat, Quoc Huy Do, Pedram Hosseyni, Ralf Küsters, Guido Schmitz, and Tim Würtele. An in-depth symbolic security analysis of the ACME standard. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, November 2021.

[7] Henry Birge-Lee. 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/, 2022.

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

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

[10] Kevin Borgolte, Tobias Fiebig, Shuang Hao, Christopher Kruegel, and Giovanni Vigna. Cloud strife: Mitigating the security risks of domain-validated certificates. In *Proceedings of the 2018 Applied Networking Research Workshop (ANRW)*, 2018.

[11] Markus Brandt, Tianxiang Dai, Amit Klein, Haya Shulman, and Michael Waidner. Domain Validation++ for MitM-resilient PKI. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 2060–2076, 2018.

[12] Markus Brandt, Haya Shulman, and Michael Waidner. Evaluating resilience of domains in PKI. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, November 2021.

[13] Baseline requirements for the issuance and management of publicly-trusted certificates. Version 2.0.1, CA/Browser Forum, August 2023.

[14] CA/Browser Forum. 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/, 2024.

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

[16] Laurent Chuat, Cyrill Krähenbühl, Prateek Mittal, and Adrian Perrig. 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)*, 2022.

[17] Laurent Chuat, Markus Legner, David Basin, David Hausheer, Samuel Hitz, Peter Müller, and Adrian Perrig. *The Complete Guide to SCION*. Springer International Publishing, 2022.

[18] Grace H. Cimaszewski, Henry Birge-Lee, Liang Wang, Jennifer Rexford, and Prateek Mittal. How effective is Multiple-Vantage-Point domain control validation? In *Proceedings of the USENIX Security Symposium*, pages 5701–5718, August 2023.

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

[20] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, IETF, May 2008.

[21] Tianxiang Dai, Philipp Jeitner, Haya Shulman, and Michael Waidner. The hijackers guide to the galaxy: Off-Path taking over internet resources. In *Proceedings of the USENIX Security Symposium*, pages 3147–3164, August 2021.

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

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

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

[25] C. Evans, C. Palmer, and R. Sleevi. Public Key Pinning Extension for HTTP. RFC 7469, IETF, April 2015.

[26] Jens Frieß, Haya Schulmann, and Michael Waidner. Crowdsourced distributed domain validation. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, pages 318–325, 2024.

[27] Artyom Gavrichenkov. Breaking HTTPS with BGP hijacking. In *BlackHat USA*, 2015.

[28] R. Gieben and W. Mekking. Authenticated Denial of Existence in the DNS. RFC 7129, IETF, February 2014.

[29] Yossi Gilad, Avichai Cohen, Amir Herzberg, Michael Schapira, and Haya Shulman. Are we there yet? on RPKI's deployment and security. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS)*, 2017.

[30] P. Hallam-Baker and R. Stradling. DNS Certification Authority Authorization (CAA) Resource Record. RFC 6844, IETF, January 2013.

[31] P. Hallam-Baker, R. Stradling, and J. Hoffman-Andrews. DNS Certification Authority Authorization (CAA) Resource Record. RFC 8659, IETF, November 2019.

[32] Elias Heftrig, Haya Shulman, and Michael Waidner. Downgrading DNSSEC: How to exploit crypto agility for hijacking signed zones. In *Proceedings of the USENIX Security Symposium*, pages 7429–7444, August 2023.

[33] Scott Helme. Using security features to do bad things. https://scotthelme.co.uk/using-security-features-to-do-bad-things/, 2016.

[34] Scott Helme. HPKP is no more! https://scotthelme.co.uk/hpkp-is-no-more/, 2020.

[35] Tomas Hlavacek, Philipp Jeitner, Donika Mirdita, Haya Shulman, and Michael Waidner. Stalloris: RPKI downgrade attack. In *Proceedings of the USENIX Security Symposium*, pages 4455–4471, August 2022.

[36] Tomas Hlavacek, Haya Shulman, Niklas Vogel, and Michael Waidner. Keep your friends close, but your routeservers closer: Insights into RPKI validation in the Internet. In *Proceedings of the USENIX Security Symposium*, pages 4841–4858, August 2023.

[37] P. Hoffman. DNS Security Extensions (DNSSEC). RFC 9364, IETF, February 2023.

[38] P. Hoffman and J. Schlyter. The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA. RFC 6698, IETF, August 2012.

[39] Joona Hoikkala. Acme-DNS. https://github.com/joohoi/acme-dns, 2024.

[40] Austin Hounsel. *Measuring the Feasibility of DNS Privacy and Security*. PhD thesis, Princeton University, 2022.

[41] Geoff Huston. Some bad news for DANE (and DNSSEC). https://blog.apnic.net/2019/05/27/dns-oarc-30-bad-news-for-dane/, 2019.

[42] Peter Kacherginsky. Celer Bridge incident analysis. https://www.coinbase.com/blog/celer-bridge-incident-analysis, 2022.

[43] Tiffany Hyun-Jin Kim, Lin-Shung Huang, Adrian Perrig, Collin Jackson, and Virgil Gligor. Accountable key infrastructure (AKI): A proposal for a public-key validation infrastructure. In *Proceedings of the International Conference on World Wide Web*, May 2013.

[44] O. Kolkman, W. Mekking, and R. Gieben. DNSSEC Operational Practices, Version 2. RFC 6781, IETF, December 2012.

[45] Mikołaj Kowalski and Wojciech Mazurczyk. Toward the mutual routing security in wide area networks: A scoping review of current threats and countermeasures. *Computer Networks*, 230, July 2023.

[46] H. Landau. Certification Authority Authorization (CAA) Record Extensions for Account URI and Automatic Certificate Management Environment (ACME) Method Binding. RFC 8657, IETF, November 2019.

[47] B. Laurie, A. Langley, and E. Kasper. Certificate Transparency. RFC 6962, IETF, June 2013.

[48] B. Laurie, G. Sisson, R. Arends, and D. Blacka. DNS Security (DNSSEC) Hashed Authenticated Denial of Existence. RFC 5155, IETF, March 2008.

[49] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A research-oriented top sites ranking hardened

against manipulation. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS)*, February 2019.

[50] M. Lepinski and K. Sriram. BGPsec Protocol Specification. RFC 8205, IETF, September 2017.

[51] Bingyu Li, Jingqiang Lin, Fengjun Li, Qiongxiao Wang, Qi Li, Jiwu Jing, and Congli Wang. Certificate transparency in the wild: Exploring the reliability of monitors. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 2505–2520, 2019.

[52] Simon Meier, Benedikt Schmidt, Cas Cremers, and David Basin. The TAMARIN prover for the symbolic analysis of security protocols. In *Computer Aided Verification*, pages 696–701. Springer Berlin Heidelberg, 2013.

[53] Donika Mirdita, Haya Schulmann, and Michael Waidner. SoK: An introspective analysis of RPKI security, 2024.

[54] National Institute of Standards and Technology (NIST). NIST RPKI monitor. https://rpki-monitor.antd.nist.gov/, 2024.

[55] M. Nystrom and B. Kaliski. PKCS #10: Certification Request Syntax Specification Version 1.7. RFC 2986, IETF, November 2000.

[56] Shivan Sahib. Delegated domain verification. https://blog.apnic.net/2023/07/03/delegated-domain-verification/, 2023.

[57] B. Schwartz, M. Bishop, and E. Nygren. Service binding and parameter specification via the dns (svcb and https resource records). RFC 9460, RFC Editor, November 2023.

[58] R.B. Shoemaker. Automated Certificate Management Environment (ACME) TLS Application-Layer Protocol Negotiation (ALPN) Challenge Extension. RFC 8737, IETF, February 2020.

[59] Yixin Sun, Maria Apostolaki, Henry Birge-Lee, Laurent Vanbever, Jennifer Rexford, Mung Chiang, and Prateek Mittal. Securing internet applications from routing attacks. *Communications of the ACM*, 64(6):86–96, May 2021.

[60] Ewa Syta, Iulia Tamas, Dylan Visher, David Isaac Wolinsky, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ismail Khoffi, and Bryan Ford. Keeping authorities "honest or bust" with decentralized witness cosigning. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, May 2016.

[61] Pawel Szalachowski, Stephanos Matsumoto, and Adrian Perrig. PoliCert: Secure and flexible tls certificate management. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, November 2014.

[62] tojens. Making DoH Discoverable: Introducing DDR. https://techcommunity.microsoft.com/blog/networkingblog/making-doh-discoverable-introducing-ddr/2887289, 2021.

[63] Dan Wendlandt, David G. Andersen, and Adrian Perrig. Perspectives: Improving SSH-style host authentication with multi-path probing. In *Proceedings of the USENIX Annual Technical Conference (ATC)*, June 2008.

[64] Sara Wrótniak, Hemi Leibowitz, Ewa Syta, and Amir Herzberg. Provable security for PKI schemes. Cryptology ePrint Archive, Paper 2019/807, May 2019.

[65] Matthias Wählisch, Robert Schmidt, Thomas C. Schmidt, Olaf Maennel, Steve Uhlig, and Gareth Tyson. RiPKI: The tragic story of RPKI deployment in the web ecosystem. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, volume 4, pages 1–7, November 2015.

[66] Jiangshan Yu, Vincent Cheval, and Mark Ryan. DTKI: A new formalized PKI with verifiable trusted parties. *The Computer Journal*, 59(11):1695–1713, July 2016.

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 [13] 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 6) and authoritative nameservers (many authoritative DNS providers offer DNSSEC). However, with the large degree of centralization in the DNS hosting of PKI domains [18], a significant portion of domains could benefit from `authenticated DNS lookups` via 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 [19, 24]

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 [57] 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 [62]. 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 a the traditional DNSSEC deployment model and only requires participation in DNSSEC by nameservers not individual domains.

B Vulnerabilities from the Lack of Declarative Security

A concrete example of a vulnerability that can emerge with lack of declarative security stems from the use of CNAME redirections for automated DNS-based domain control validation challenges, such as the ACME-DNS service [39]. In this scenario, a domain creates a CNAME record for the ACME dns-01 "_acme-challenge" subdomain pointing to another subdomain provisioned in the challenge automation service's domain zone (e.g., "acme-dns.io"). Even if the customer domain has a DNSSEC-signed CAA record stipulating "validationmethods=dns-01", the full cryptographic security of the validation is dependent on the DNSSEC signing of the delegated ACME dns zone—and "acme-dns.io" is not DNSSEC signed. Thus, the last hop TXT record lookup to the client's "acme-dns.io" subdomain is unsigned and vulnerable to network attacks. If a domain instead declaratively states its desire for cryptographic DV, the CA can act upon this to not accept validation over any plaintext channels.

Table 4: CAA records and properties achieved.

CAA Extension	Supports non-ACME CAs	Declarative Security	Enables Global Properties Across CAs
RFC 8657	○	○	○
"security" CAA tag	●	●	●

Table 5: 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)	✗	✗

C Formal Model Details

We verify the security properties of our design, described in Section 4.1, by modeling the behavior and capabilities of certain entities as outlined below.

Model. We create a formal model using the security protocol verification tool Tamarin [52] based on the relevant protocol-specific standardization documents, such as IETF drafts and CA/Browser Forum documents [13]. We use the Tamarin modeling tool as it is designed to prove properties of network security protocols, has a built-in Dolev-Yao attacker, and has been used in prior web PKI research, such as ARPKI [5] and DTKI [66]. Compared to previous formal models, e.g., the RFC-based model of ACME protocol [6], 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 [64]. 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.

Our Tamarin model considers four types of actors: domain owners controlling second-level domains and subdomains, DNS zone owners controlling parent domains (e.g., TLDs), certificate-issuing CAs [13], and a Dolev-Yao adversary able to intercept and inject arbitrary messages in the network. We model the following security-relevant assets: asymmetric TLS and ACME account key pairs of domain owners [4], and DNSSEC zone signing key pairs of zone owners. Our model includes all messages that are relevant for the security properties achieved by cryptographic DV: DNS CAA records [31, 46] to restrict TLS certificate issuance, DNS NSEC(3) records to provide proof of non-existence of DNS

Table 6: 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 protected domain (per the Server Certificate 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 is a secure CA that 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 [3, 28, 48], and messages related to TLS certificate issuance, such as Certificate Signing Requests (CSR) [55] and X.509 certificates [20]. The model is described in around 2000 lines of Tamarin code.

We assume that domain owners either specify a caOwnerID 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.

Proofs. In Tamarin, properties are proven through lemmas, which are logical statements over possible protocol executions, called protocol traces. For example, the first security property of our proposed system (secure issuance) is partly described in the *universal quantification* Lemma 1: In all protocol traces where a caOwnerID is accepted by the secure CA to issue a certificate, the caOwnerID 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. Furthermore, we show that our model allows for the issuance of fraudulent certificates if entities do not properly secure their policies through DNSSEC, do not specify either caOwnerID or a secure validation method, or do not specify the critical flag in their policy. Successful protocol executions and attacks are shown through *existential quantification* lemmas, e.g., an adversary injecting

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

```
lemma OnlyLegitimateIDAcceptedForVictimDomain:
 All ID #i.
   // If the secure CA accepts an owner ID,
   CaOwnerIDAccepted('domain', ID) @i
  ==>
   // then it was either the legitimate ID, or
   (Ex #j. IsLegitimateCaOwnerID(ID) @j) |
   // the domain owner's DNSSEC key
   // was revealed to the adversary
   (Ex #k. DnssecKeyRevealed('domain') @k)
```

a fake CAA record through a BGP hijack to obtain a fake certificate for a non-DNSSEC domain.

Through a combination of such lemmas, we cover our security properties and successfully prove that, given the threat model described in Section 3.1, all security properties hold under the assumptions described in Appendix C.

Assumptions. 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.

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. Furthermore, following our threat model, we assume that the domain owner and all parent domain owners are benign and that all domain owners have authentic communication channels to their parent domain owners for updating their DNS(SEC) records.

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 [13], 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 [28]. 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 [44]) and instead treat revoked keys as non-existent keys.

D A look at top cryptographic DV domains

The 74.8K domains who are early adopters of cryptographic DV include many popular websites in the top 1M Tranco [49] site ranking (generated on 1 May 2024 [2]) including official sites that host several popular OSes (Debian, GrapheneOS), the Bitcoin Core source code, the Tor project, the Slack messaging app, and the XMPP-based messaging app Jabber. A selection of these high-impact domains is included in Table 7. The full list of domains using cryptographic DV can be downloaded anonymously from Github.

Table 7: Details of issuance policies for top ranked domains using cryptographic DV elements.

Domain name	Ranking	Domain policy contents		
		DNSSEC signed?	Account ID?	Secure DV method?
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	✓	✓	

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