Infer, Don't Negotiate: Unilateral ML-Based Cipher Selection for DNSSEC

Stefan Schmid[*‖], Haya Schulmann[†¶], Juan Vanerio[‡**], Michael Waidner[§¶‖]
[*]TU Berlin, Germany [†]Goethe-Universität Frankfurt, Germany [‡]University of Vienna, Austria [§]TU Darmstadt, Germany
[¶]ATHENE, Germany [‖]Fraunhofer SIT, Germany [**]Austrian Institute of Technology, Austria

Abstract

Cryptographic agility, migrating to stronger algorithms, is a prerequisite for DNSSEC's long-term security, yet it is stuck: almost all signed domains use a single algorithm. Cipher-suite negotiation needs changes at *both* ends and hence has still not been deployed. We reframe the deadlock as *inference* rather than negotiation: the capabilities of a resolver can be inferred from its query. Our key observation is that a nameserver can predict the strongest algorithm the resolver accepts from queries and serve that *unilaterally*, with no resolver-side change. The prediction is performed by a *capability oracle*, which can be instantiated with any method that predicts which algorithms a resolver validates, e.g., a learned classifier. We implement the oracle using an off-the-shelf random forest, and show experimentally that it predicts per-algorithm support at Matthews Correlation Coefficient (MCC) ≥ 0.88. On queries the forest has not previously seen, serving Ed448 fails for 1% of queries from validating resolvers instead of 54.5% with a naive baseline, i.e., serving Ed448 to every validating resolver, with no oracle.

Inference matches a plain dual-signed zone's availability while shrinking responses and reducing the attack surface and opening a migration path. Our proposal is downgrade-resilient independent of predictor accuracy. The ideas in this research extend to other migrations that lack negotiation.

CCS Concepts

• **Security and privacy** → **Network security**; • **Networks** → *Network protocols*; • **Computing methodologies** → *Machine learning*.

Keywords

DNSSEC, cryptographic agility, algorithm negotiation, post-quantum migration, machine learning



This work is licensed under a Creative Commons Attribution 4.0 International License.
HotNets '26, Salt Lake City, UT, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-3029-0/2026/11
https://doi.org/10.1145/3847290.3859949

ACM Reference Format:
Stefan Schmid, Haya Schulmann, Juan Vanerio, and Michael Waidner. 2026. Infer, Don't Negotiate: Unilateral ML-Based Cipher Selection for DNSSEC. In *The 25th ACM Workshop on Hot Topics in Networks (HotNets '26), November 16–17, 2026, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3847290.3859949

1 Introduction

DNSSEC secures Domain Name System (DNS) by signing resource records, so resolvers can verify that answers are authentic [4, 40]. Its long-term security depends on *cryptographic agility*, retiring weak algorithms and adopting strong ones as cryptanalysis advances and the post-quantum transition nears [41]. DNSSEC was designed to be agile: thirteen signing algorithms are registered, and a zone owner may sign with any subset of them [43, 66].

Agility has stalled. Almost all signed zones pin a single algorithm (§3) [16], because signing with a stronger algorithm that an old resolver does not recognize can break resolution or, worse, *downgrade* the zone to unvalidated and reopen it to cache poisoning [35, 36]. The textbook remedy, cipher-suite negotiation [38, 39], needs new signaling at *both* ends, and neither proposal was standardized or deployed. The agility an operator can exercise today is bounded by the weakest resolver it is unwilling to break.

Infer, don't negotiate. A resolver already discloses its capabilities implicitly, in the bits of every query it sends (its source address and network, its EDNS options and advertised buffer size). A nameserver need not be *told* which algorithms a resolver supports; it can *predict* them. Agility, long treated as a coordination problem, becomes a *server-side inference* problem one party can solve unilaterally.

Contributions. We: (1) reframe DNSSEC algorithm agility as a unilateral inference problem deployable without requiring changes on the resolvers (§2-§4); (2) design our mechanism: dual-signed zones, a stripping middlebox, and a conservative-algorithm floor with a source-authentication gate that make downgrade resilience independent of the *capability oracle* predicting what each resolver validates (§4, §6); (3) instantiate the oracle with off-the-shelf trees and test against behavioral ground truth: it reproduces direct tests of four public operators, and on client sessions it has never seen it reduces the failures of serving Ed448 from 54.5% to

	RSA				ECDSA		EdDSA	
Population	5	7	8	10	13	14	15	16
Open resolvers	99.9	100	100	100	98.4	98.4	85.1	21.8
Ad-network	99.6	99.6	99.7	99.7	97.5	97.4	70.8	33.9

Table 1: Algorithm support (% of validators), collapses at modern EdDSA algos - our targets for prediction.

1% (§5.2); (4) evaluate on ∼10M resolvers showing that *availability parity is the ceiling*, and show what filtering offers below it (smaller responses, a reduced attack surface, a migration path) (§5); (5) show that our proposal applies beyond DNSSEC, and provide directions for future work (§7).

2 The Agility Deadlock

DNSSEC has barely exercised its algorithm choice.

Negotiation needs both ends. DNSSEC, unlike TLS or IPsec, has no cipher-suite negotiation: the resolver does not announce which algorithms it validates, and the nameserver cannot ask. Proposals to add it introduce changes on resolvers *and* nameservers [38, 39]. Even the EDNS *signal* of understood algorithms [19] saw little deployment, and RFC 7696 presumes endpoints that negotiate [41]. Negotiation would also need a downgrade defense: an EDNS signal is unauthenticated, so an on-path attacker could rewrite it to downgrade to a weaker algorithm.

"Sign with everything" is not free. Operational guidance has zones sign with *all* advertized algorithms so any resolver finds one it can validate [65]. Multi-algorithm zones inflate responses, expose to fragmentation attacks [37], and multiply the resolver's cryptographic work [33]. A 2025 cache-flushing attack abuses oversized multi-algorithm signatures to cut a resolver's serving capacity to as little as 6% [6].

Stronger can mean weaker. An unrecognized algorithm makes on the order of 45% of resolvers stop validating and fall back to insecure resolution [34], and an adversary can exploit the same transition to downgrade a signed zone to *insecure* [35]. Operators therefore stay on the algorithm the resolvers already handle, without a way to move before the rest of the Internet does.

3 How Stuck Is Agility?

Our October 2026 census of DNSSEC deployment, run over DNS-over-HTTPS shows:

Domains barely move. Of the Tranco top-20k only 9.99% of domains are signed (DS and DNSKEY; in March 2024 we found 6.77% of the top-1M), and of the 1,437 top-level domains 94.0% are signed (93.9% in 2024). Almost no zone exercises agility: 98.7% of signed top-20k domains and 96.8% of signed TLDs publish keys of a *single* algorithm. Ed25519 appears in 0.3% of signed domains and three TLDs, Ed448 in none, while legacy RSA/SHA-1 (5, 7) persists in 3% of signed domains and three TLDs despite long-standing guidance to retire it [53, 66], now a formal deprecation [30]. In addition, 39 TLDs now publish DS and DNSKEY records for both RSASHA256 and ECDSAP256 (dual-signed configuration).

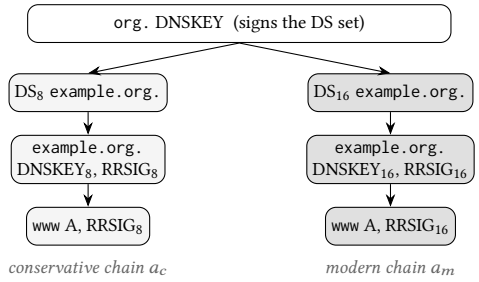


Figure 1: Dual-signed zone exposes one chain of trust per algorithm (conservative a_c=8 and modern a_m=16). Each resolver follows the strongest chain it supports.

Resolvers are the obstacle. 27.19% of open resolvers from port scans validate DNSSEC, as do the resolvers of 31.02% of web clients we reached through an ad network, in line with APNIC's ad-based measurement [42]. Among validating resolvers, all support the RSA and ECDSA algorithms, but only 85.09% support Ed25519 (15) and just 21.78% support Ed448 (16) (Table 1). This is the main obstacle to agility, and the motivation for a per-resolver decision.

4 Infer, Don't Negotiate

Our proposal replaces the missing negotiation with prediction on the authoritative side. A zone is signed with two algorithms. A *capability oracle* predicts, per query, which algorithm the resolver can validate and the nameserver responds with that one. Our unilateral inference guarantees a conservative selection, hence even a wrong answer does not expose the resolver to a downgrade below a_c. Capability oracle can be instantiated with an off-the-shelf forest as in our earlier poster [32] and passive fingerprinting [2, 5, 29, 68].

Dual-signed zones. The operator signs the zone with a *conservative* algorithm a_c that validating resolvers must support (e.g., RSASHA256 or ECDSAP256, numbers 8/13 [31, 66]) and a *modern* algorithm a_m they may not (e.g., Ed25519 or Ed448, 15/16), with DNSKEY, RRSIG and a parent DS record (Fig. 1). A resolver that supports a_m validates the modern chain, one that does not falls back to a_c, and only one supporting *neither* fails, which the choice of a_c makes rare.

The middlebox. The prediction and selection are performed by a reverse-proxy *middlebox* between the backend nameserver and the Internet. The backend is unmodified and answers every query with signatures for *both* a_c and a_m; the middlebox asks the oracle whether the querier supports a_m and on egress *strips* the de-selected algorithm's RRSIGs, forwarding a response signed only with the selected a_s. All logic

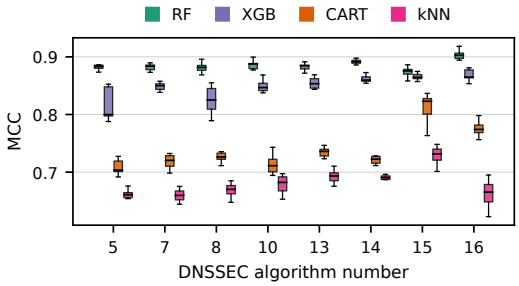


Figure 2: MCC per algo. and model over 10 query folds. RF leads for every algorithm, XGBoost close behind, with a wide margin over the non-ensemble baselines.

Algorithm	Model	TN	TP	FP	FN	P	MCC
rsasha512	RF	52%	43%	4%	2%	44%	0.89
ecdsap256sha256	RF	52%	43%	4%	2%	44%	0.89
ecdsap384sha384	RF	52%	43%	4%	2%	45%	0.89
ed25519	RF	56%	38%	5%	1%	39%	0.88
ed448	RF	82%	14%	2%	1%	16%	0.89
rsasha1	RF	51%	44%	4%	1%	45%	0.89
rsasha1nsec3sha1	RF	51%	43%	4%	2%	45%	0.89
rsasha256	RF	51%	44%	4%	2%	45%	0.88

Table 2: Best classifier per algorithm (test fold; cells are % of test instances; positive = the resolver validates the algorithm). P is the positive prevalence in the labeled query mix (TP+FN before rounding). MCC $\geq$ 0.88 and FP$\leq$5% throughout on query-level folds (unseen sessions: Table 3), including the rare Ed25519/Ed448. Rows that coincide have nearly equal support prevalence in the labeled mix and round to identical cells.

is in the middlebox, which can be deployed in a domain with no change to zone-signing tools, nameserver or resolvers.

The response. We keep all DS and DNSKEY records and RRSIGs over the DNSKEY set. For other record types we keep only the RRSIG of the selected algorithm a_s when the invariants of §6 permit stripping for that querier, and the conservative RRSIG otherwise. Filtering can apply to the queried records only (*sigt*) or also to DNSKEY queries (*sigk*), and the zone can use a common signing key (*csk*) or split KSK/ZSK (*ssk*). Filtering DNSKEY responses as well (sigk) costs up to 14 points of query success when a_m is Ed448 (§5.3), which is why (I1) of §6 leaves DNSKEY material unfiltered. The key scheme changes query success by at most 3 points.

5 Does It Work?

We demonstrate feasibility experimentally, showing that support can be predicted, and availability is preserved.

5.1 Predicting support

Data and models. We label query features with ground-truth support from three resolver populations: nine lab stacks (BIND9, Unbound, PowerDNS, Knot Resolver, five Windows Server variants), open resolvers from port scans, and 3,526 validating resolvers of web clients reached through an ad network [42]. Each population resolves names under domains we control, one zone signed exclusively with each of the eight algorithms in RFC 8624 [66]. A query is labeled supported only when the resolver validated that zone's algorithm, and insecure fallback counts as unsupported. Each test zone carries deliberately invalid signatures, so a resolver that validates its algorithm answers SERVFAIL and one that does not, returns the record. Features come from the packets our nameservers receive: the IP TTL, DF bit, ToS and length, the source address, its routed prefix, origin AS and region, UDP length and source port, the DNS flags DO, CD and AD, the QTYPE and the QNAME's 0x20 case, the EDNS version, buffer size, option count and the presence of COOKIE and ECS options, and per-packet IP ID, DNS ID and checksums. Predicting support is tabular classification over mixed categorical and numeric fields. We use off-the-shelf tree ensembles, a standard choice for such data [28], because they need little tuning and expose feature importances, which §5.2 uses to check what the oracle relies on. A domain operator is free to choose any other classifier (§4). We compare Random Forests [9] and XGBoost [14], so agreement between bagging and boosting shows the result is not an artifact of one method, against a single CART tree [10], k-NN and a majority-class dummy. We evaluate under 10-fold cross-validation and report the MCC, which, unlike accuracy or F_1, stays near zero for a constant predictor under class imbalance [15]. We apply SMOTE [13] oversampling and the median-importance feature selection to the training fold only, so neither can leak into the score. Each forest has 100 trees, the share of features tried per split (0.3 to 0.9) is set by a randomized search with nested 3-fold cross-validation, and features below the median importance are dropped. The errors have different implications: positive means that the resolver validates the modern algorithm a_m. A false positive serves a_m to a resolver that cannot validate it, hence it returns SERVFAIL. A false negative serves the conservative a_c to a resolver that could have validated a_m: the query succeeds, but the migration to the stronger algorithm failed for that resolver. The decision threshold maximizes MCC on the training fold, which weighs both errors. Because a false positive affects availability, a deployment should raise the threshold until failures fall below a budget, which we evaluate in §5.2 with a budget of 1%.

Results. Random Forests are best for every algorithm, XGBoost a close second, with a wide gap to the non-ensemble baselines (Fig. 2). The best per-algorithm classifiers reach MCC$\geq$ 0.88 on query-level folds with false positives at most

5% of test queries (Table 2), including the rare Ed25519 and Ed448. A false positive harms only resolvers that validate a_c but not a_m. A non-validating resolver simply ignores signatures. The positive prevalence (column P) is a property of the query mix, which over-represents validators: validating resolvers send 44% of the queries to our RSASHA256 control zone, while 27-31% of resolvers validate (§3).

5.2 Ground Truth and Unseen Clients

A resolver's software family is leaked in its queries, which is what passive fingerprinting exploits [5, 29], but whether it validates Ed448 also depends on build options and its crypto library, so two resolvers that send identical queries can differ. We therefore test the oracle against *behavioral* ground truth, what a resolver demonstrably validates.

Known operators. For Google, Cloudflare, Cisco OpenDNS and Neustar, the forest's out-of-fold predictions agree with direct tests for RSASHA256, Ed25519 and Ed448 in all 12 cases. Re-tested over DNS-over-HTTPS in October 2026, Google and Cloudflare validate nine Ed25519-only zones but treat the `.no` registry's Ed448 test zone as insecure (Cloudflare: EDE 1), while Cisco OpenDNS validates both, as the forest predicts. Since these operators appear in training, both checks show that our labels are right and that the forest has learned these operators. To show that it predicts resolvers it has not seen, we measure with the grouped folds: on unseen queries MCC is 0.70 for Ed25519 and 0.47 for Ed448 (Table 3), and with a 1% failure budget serving Ed448 fails for 1% of queries from validating resolvers.

What the forest uses. Permutation importance reveals which features influence the forest's predictions. For Ed25519 the strongest feature is the EDNS buffer size (0.35), a default that resolver software sets and has changed across versions. Summed over the DNS and EDNS fields the drop is 0.38, against 0.27 for the identity features, and the IP TTL, which the operating system sets, adds 0.15. For Ed448 the identity features dominate (0.74, the AS alone 0.56), followed by the IP TTL (0.23) and the buffer size (0.16). The forest thus tracks the software where support follows the software, as for Ed25519, and falls back on which networks run capable resolvers where support depends on build options, as for Ed448. The randomized per-packet fields carry 0.08, mostly through the IP checksum, computed over the source address, TTL and IP ID. Retrained without these fields, the forest still reaches MCC 0.85-0.88 on query-level folds (0.87-0.90 with them in the same re-run) and does better on unseen sessions, 0.72 for Ed25519 and 0.58 for Ed448 against 0.70 and 0.47. These fields help the trees memorize networks already seen.

What each split measures. Query-level folds split individual queries at random, so a test query usually comes from a resolver the forest has already seen: 94%–96% of test queries come from source addresses that also appear in training. These folds therefore measure how well the forest predicts known resolvers. That is the situation of a nameserver that retrains regularly on its own traffic, as long as most resolvers that query it keep coming back. Grouped folds measure networks the forest has not seen. Every query name carries the token of the client session that triggered it, so we split by session, by source address and by routed prefix. No group contributes to both training and test (Table 3). One resolver can use many addresses, so address-disjoint folds are optimistic; a held-out prefix removes whole kinds of resolvers, so prefix-disjoint folds are pessimistic, and sessions share resolvers (65-69% of test queries still come from addresses seen in training). On unseen sessions MCC is 0.61-0.65 for RSA and ECDSA, 0.70 for Ed25519 and 0.47 for Ed448, inside the range the two bounds span. The gap to the query-level score bounds from above how much the forest relies on networks it has seen. With few large groups the folds also vary (standard deviation 0.18 for Ed448 on sessions). On new sessions the identity features (address, prefix, AS, region) alone do as well as all features or better, because large resolvers recur across clients. Without them, Ed25519 keeps MCC 0.64, a signal carried by the queries and not by who sends them.

	Ed25519	Ed448
MCC, query-level folds	0.89	0.90
MCC, unseen source addresses	0.70	0.58
MCC, unseen sessions	0.70	0.47
without identity features	0.64	0.38
identity features only	0.69	0.57
MCC, unseen routed prefixes	0.49	0.26
Failures, a_m served to all	9.2%	54.5%
Failures / reach, MCC threshold	2.1% / 83%	2.1% / 56%
Failures / reach, 1% budget	1.0% / 46%	1.0% / 49%

Table 3: RF as oracle on sessions, addresses and prefixes absent from training (10 grouped folds per split). Failures: share of queries from RSASHA256-validators that get a_m but cannot validate it. Reach: share of the validators of a_m that receive it.

Serving costs. MCC weighs both errors alike, but only a validator of a_c can fail on a modern answer. Among queries from sessions that validate RSASHA256 (most of their control queries fail on the invalid zone), serving Ed448 to everyone fails for 54.5%. The forest, on sessions it has never seen, fails for 2.1% and reaches 56% of the validators that support Ed448. A failure budget of 1%, set on the training folds and frozen before testing, holds on average over the test folds (1%, standard deviation 1.1 points) and still reaches 49% of them; within RSASHA256-validating sessions, 95% of the queries it serves Ed448 to, originate from resolvers that validate it.

Algorithm	RRSIG sig.	DNSKEY key
RSASHA256 (8)	256	260
ECDSAP256 (13)	64	64
ECDSAP384 (14)	96	96
Ed25519 (15)	64	32
Ed448 (16)	114	57
ML-DSA-44	2420	1312
Falcon-512	752	897
SPHINCS$^+$-128s	7856	32

Table 4: DNSSEC sizes (bytes). The lower block gives post-quantum (pq). Stripping the de-selected RRSIG saves its signature size per RRset: a few bytes for an all-ECC pair, hundreds for an RSA a_c, kbytes for pq a_m.

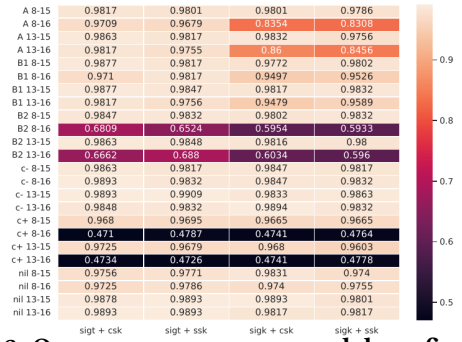


Figure 3: Query success across model configurations (rows) and key scheme/filter placement (columns). Most configurations are near the ~0.98 baseline; *sigk* and aggressive modern-only selection degrade.

5.3 Feasibility Evaluation on the Internet

We measured 10M open-resolver addresses, resolving names under domains we control, and compared query-success rates across signing strategies and filter placements (Fig. 3). Rows combine three trained models (A, $B1$, $B2$). A uses the query features of §5.1, $B2$ only features that encode each query's membership in clusters of similar queries, and $B1$ uses both, a constant-positive baseline $c+$ (always a_m; an upper bound on false-positive damage), a constant-negative $c-$, and *nil*, which serves both algorithms unfiltered (the same zone *without* our mechanism). Columns vary the a_c/a_m pair, the key scheme (csk/ssk), and the filter placement (sigt/sigk). An inference-driven zone reaches up to 98.77% query success, close to that of an *unsigned* domain (99.08%) and an RSASHA256-only domain (98.64%). The same zone without the filter (*nil*) reaches 98.93%, within run-to-run variation. *nil* is the availability ceiling, determined by capable resolvers that can validate dual-signed zones without our mechanism.

What does filtering offer beyond *nil*? Notably a smaller response: each signed RRset carries one signature instead of two. *Size*: dropping an RSASHA256 signature frees 256 bytes per RRset against the UDP/MTU limit that drives fragmentation and amplification [20, 64], and a post-quantum a_m frees kilobytes (Table 4), often the difference between a UDP answer and a forced TCP fallback. *Attack surface*: the resolver holds one algorithm's material, so it cannot be driven into KeyTrap-style downgrade [33]. The "sign-with-everything" zone we measure has far worse availability (91.28%) and the slowest latency. *Migration*: *nil* must always serve a_c, whereas inference makes the per-resolver decision, so the operator can count served-modern answers and the failures they cause, and retire a_c once probes of the resolvers show that they too validate a_m (§6). Inference is the only option that provides availability, shrinks the response, *and* yields a retirement signal. The placement and scope of inference affect the result: DNSKEY material must stay unfiltered (§6).

Per-query inference adds little delay: response times track the unsigned and single-algorithm baselines (most within ~1 s). Only the sign-with-everything zone is slower.

5.4 Caching and Shared Resolvers

DNS uses shared caches: a validator replays the RRset it receives to other clients. In the common topology the querier *is* the validator, serving non-validating stubs (CD= 0), so the resolver we predict is the one the cached entry must match. The risk is a *forwarder* that caches without validating, in front of *heterogeneous* downstream validators with weaker support. Such structures are common [50, 58]: > 90% of forwarding resolvers sit behind under 5% of the resolvers [67], and most resolvers cache DNSSEC records without validating [16, 49]. The invariants of §6 cover this case: predict on the node that sets DO and validates, bias forwarder-like queriers to a_c, use EDNS Client Subnet [17] only as a tie-breaker, and keep the conservative RRSIG on every RRset a downstream validator might re-serve, including the NSEC/NSEC3 proofs [23].

6 Security Against Downgrade

Negotiated protocols are the classic target of downgrade attacks [7, 8], and TLS detects tampering by authenticating the handshake transcript [52]. Inference has no transcript: its inputs are unauthenticated query fields. We therefore state a floor guarantee against DNSSEC's standard on-path adversary and a weak off-path adversary, which is not based on input authenticity.

Spoofable inputs. The oracle's inputs are unauthenticated, and even an off-path adversary can forge DNS packets [37, 47], so an attacker can craft queries that steer the prediction [3, 54]. We make that steering harmless.

What an attacker gains. A *false positive* (modern predicted but unsupported) yields a response the resolver cannot validate, SERVFAIL: DoS an on-path attacker can already cause.

Prediction state must therefore be uncached or unforgeable, or one spoofed query could make that DoS persistent. A *false negative* serves the conservative a_c, which validating resolvers must support. Neither drops the zone *below* a_c, the dual of the resolver-side downgrade of [35], where the adversary strips DNSSEC to *insecure*.

A conditional floor guarantee. Invariants: **(I1)** The parent always publishes a valid conservative DS and inference is *never* applied to DS or DNSKEY material, so the DNSKEY set keeps the conservative key and a conservative RRSIG (this is why filtering DNSKEY queries, *sigk*, collapses availability, §5). **(I2)** Every signed RRset a validator may see (answers, wildcards, and NSEC/NSEC3 denial proofs, *including those re-served from a shared cache*, §5.4) keeps at least the conservative RRSIG unless the recipient is predicted to be both modern-safe *and* a terminal validator with homogeneous cache scope. A re-serving forwarder, or any querier we cannot confidently place, gets the conservative RRSIG. **(I3)** a_c is a currently-strong mandatory-to-implement algorithm. **(I4)** Prediction state is not settable by a forged query. **(I5)** The conservative RRSIG is stripped only for a source whose address has been confirmed by a completed DNS Cookie or TCP exchange [21], so an off-path spoofed query [47] cannot trigger a downgrade. The same TC-to-TCP escalation POPS uses against poisoning shows this gate is server-side and unilateral [1], at the cost that the first, unconfirmed query from a source is served conservatively.

In invariant (I2), mixing NSEC and NSEC3 denial within a zone is insecure, and five resolvers and a forwarder cached the misleading proof [69]. Placement, whether a querier validates itself, is probed, without relying on spoofable query bits: a validator resolves a correct name and rejects deliberately misconfigured ones [49]. Its failure modes (topology drift, anycast, hidden downstreams) are treated conservatively and raise only the fallback rate.

Safety does not rely on prediction accuracy. Because (I1)-(I5) hold for *every* prediction of the oracle, a wrong, drifted, or adversarially evaded prediction costs at most a fallback to a_c, which impacts only availability. Evasion by input shaping [3, 54] thus stays inside the DoS threat model we already accept, and the predictor need not be adversarially robust for the mechanism to be *safe*. Its accuracy is an *availability* property, which §5.2 quantifies on unseen client sessions.

Claim. If (I1)-(I5) hold and validators fail closed on modern-only answers they cannot verify, then for every adversary strategy the forced state is availability loss or fallback to a_c, but not below a_c: the mechanism is *downgrade resilient* [8]. The fail-closed premise is what the DNSSEC standard requires. A modern-only answer is an ordinary single-algorithm signed answer, and a validator that cannot build a chain must return SERVFAIL, not insecure [4]. A resolver drops to insecure only when it supports *none* of a zone's DS algorithms [65]. (I1) always publishes a conservative DS it does support, so a missing a_c RRSIG is bogus, not insecure, which resolves the tension with §2. RFC 6840's mandatory-algorithm rules bind signers, not validators, so no validator rejects an answer for a missing algorithm's signature [65]. Single-RRSIG validation is also the common path: about 99% of signed zones use one algorithm (§3). What remains is to confirm that no resolver version regresses from this behavior, including under aggressive NSEC re-use [23].

The fallback rate is bounded by ecosystem base rates. With ~27-31% of resolvers validating (§3) and 85% of those supporting Ed25519 (far fewer Ed448, Table 1), at most ~23% of queries can be served Ed25519 today and ~6-11% Ed448; the placement and authentication gates only lower these bounds. Every fallback is safe, and the served-modern share rises with adoption. That share is not the signal to retire a_c: an oracle that answers "modern" to everyone maximizes it, and one that answers it only to capable resolvers sees no failures while the others stay on the floor. Retirement requires evidence about the resolvers still served a_c: probes with test zones signed with a_m alone, as in §3.

Attacks on the model. Membership inference [60] and model extraction [27] expose no signing secret. The concern is *data poisoning* [26], an adversary corrupting the features or labels we collect. By the decoupling above, even a poisoned model cannot force a state below a_c, so poisoning is an availability attack. De-selection also blunts the KeyTrap [33] and cache-flushing attacks, where LAPRAD inflates a 449-byte response into 65 KB [6].

7 Generalizing Unilateral Inference

"Infer, don't negotiate" requires four preconditions (Table 5): (1) capability differs across peers, (2) negotiation is undeployable because both ends must change at once [41], (3) one party can serve several variants and select per peer, and (4) a safe default exists, so a wrong guess degrades to a floor. DNSSEC satisfies all four. In addition to DNSSEC, other protocols can also benefit from unilateral inference. We provide examples for selected systems (Table 5).

DKIM: RFC 8463 obliges DKIM verifiers to implement Ed25519, yet tells signers they "can add multiple signatures that use old and new signing algorithms" [45], DKIM's sign-with-everything [18]. A sender that signs per destination domain could keep RSA as its floor and add Ed25519 only where it predicts a capable verifier. Forwarding and mailing lists take the role of the shared caches of (I2).

Software-update repositories: when a server hands each client a signed object, the Update Framework (TUF) metadata may carry several signatures, a role requires only a

threshold of them, and TUF "is not restricted to any particular signature schemes" [12, 55]. If each algorithm's keys alone meet the threshold, a repository can sign with a classical and a post-quantum key and serve each client only the signatures it verifies. The client names its software in the User-Agent, which servers already use to "tailor responses to avoid particular user agent limitations" [22], and mirrors and CDNs take the role of the shared caches of (I2).

OCSP has standardized the pattern: a responder may select "the signature algorithm used to sign the OCSPRequest", and otherwise "a mandatory or recommended signature algorithm" [56], which is inference with a floor. An S/MIME sender can sign per recipient, since the SMIMECapabilities attribute of earlier messages lists the "signature algorithms" a peer handles [57], a hint like DAU.

RPKI would operate two parallel hierarchies, one per algorithm suite, whose products "SHOULD be stored at independent publication points" [24]. A repository Server would publish instances of each object signed with different algorithms. A relying party (RP) that fetches objects from a repository hosting both suites thus would have to download both, a cost that grows with post-quantum signatures. A server that knew which suite an RP validates could hand it only the signed objects belonging to that suite (its CA certificates, manifests, CRLs and ROAs) and the old suite, as the floor, to any RP it cannot place. RRDP recommends that RPs send a User-Agent header naming their software, since it is "useful to track capabilities of Relying Parties" [11], and HTTPS carries it. The same log gives the procedure the retirement signal it lacks: its timetable fixes the phase dates in advance, and "the timetable MUST be reissued" if "many RPs might not be ready" [24].

8 Can Hints Replace Inference?

Any hint a DNS resolver sends about its algorithms faces general obstacles. RFC 6975 lets a validating resolver list the algorithms it understands in an EDNS option (DAU) [19]. None of the 347,962 labeled queries from 9,737 resolvers in our data carries it, and the RFC rules out our use: the options "MUST NOT trigger any special processing (e.g., RRSIG filtering in responses) on the server side" [19]. *Deployment:* a hint helps only once resolvers send it by default. The resolvers that matter most are the old ones that are never updated and do not implement any newly deployed algorithm, hence they will not send it. This use case is exactly where inference is needed most. *Authentication:* DNSSEC does not sign queries, so an off-path attacker can forge a hint and an on-path one can strip or rewrite it. Further problems include privacy, e.g., the list can identify the validator; interoperability with intermediate resolvers, e.g., a forwarder sends the union of its own and its client's list, possibly with algorithms the resolver does not validate. Hints therefore leave (I1)–(I5) unchanged.

Migration	(1)	(2)	(3)	(4)
DNSSEC, RSA/ECDSA → EdDSA	✓	✓	✓	✓
DNSSEC, classical → PQ	✓	✓	✓	✓
DKIM, RSA → Ed25519	✓	✓	✓	✓
Update signing (TUF)	✓	✓	✓	✓
OCSP responses	✓	✓	✓	✓
S/MIME signatures	✓	✓	✓	✓
RPKI, RRDP repository	✓	✓	✓	✓

Table 5: Preconditions (1)-(4) of §7.

The same obstacles recur in other systems. A hint can be forged and deployment and interpretation challenges also remain: old peers send no hint, an S/MIME list is "a partial list", a User-Agent names software rather than its configuration, and every list fingerprints its sender.

9 Related Work

Agility and negotiation. Measurements document slow, skewed adoption and the hazards of changing cryptographic material at scale [16, 48]. *Negotiation* has both ends agree on an algorithm [38, 39, 41, 66], including for post-quantum DNSSEC [61]; *global measurement*, for example by the Root Canary [63], observes capability but does not act. Whereas our preliminary poster [32] explored ML-based capability prediction, this work develops a pipeline for unilateral algorithm selection, with an explicit security architecture that separates downgrade protection from prediction accuracy, while treating the capability oracle as a replaceable component that can use different inference methods or classifiers.
Post-quantum. PQ schemes fit heavier algorithms into DNS by request- and `QNAME`-based fragmentation [25, 51], on-the-fly signing [44, 62], or dual classical-plus-PQ signatures [59]; TLD-scale study reports the same pressure against the 1232-byte UDP buffer, with only Falcon-512 close to deployable [46] (sizes in Table 4). All hand every resolver the same large response, and those without a downgrade exposure need resolver-side support. Unilateral inference does not require changes at the resolver.

10 Conclusion

A decade of negotiation proposals could not move DNSSEC off its weakest algorithm. A nameserver can instead *infer* the strongest algorithm a resolver accepts and serve it unilaterally. The unilateral inference ensures agility whatever the capability oracle answers, and an off-the-shelf forest is already a usable oracle: on unseen client sessions, Ed448 reaches half of the queries from its capable validators at 1% failures. Availability stays at the dual-signed ceiling while responses shrink and a migration path opens.

Acknowledgments

This research work was supported by the National Research Center for Applied Cybersecurity ATHENE. ATHENE is funded jointly by the German Federal Ministry of Research, Technology and Space and the Hessian Ministry of Science and Research, Arts and Culture. This work has been partially funded by the FWF Austrian Science Fund, Project reference I-6653, as part of the EU CHIST-ERA-2022-SPiDDS-02 project *GRAPHS4SEC*. This work has been partially funded by University of Vienna, UniVie Doctoral School Computer Science, Vienna, Austria. Supported by the German Research Foundation (DFG), Schwerpunktprogramm SPP 2378 - ReNO-2 (grant 511099228), 2025-2029.

References

[1] Yehuda Afek, Harel Berger, and Anat Bremler-Barr. 2025. POPS: From History to Mitigation of DNS Cache Poisoning Attacks. arXiv preprint arXiv:2501.13540.

[2] John Althouse, Jeff Atkinson, and Josh Atkins. 2019. TLS Fingerprinting with JA3 and JA3S. Salesforce Engineering, https://github.com/salesforce/ja3.

[3] Giovanni Apruzzese, Hyrum S. Anderson, Savino Dambra, David Freeman, Fabio Pierazzi, and Kevin A. Roundy. 2023. "Real Attackers Don't Compute Gradients": Bridging the Gap Between Adversarial ML Research and Practice. In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. arXiv:2212.14315.

[4] Roy Arends, Rob Austein, Matt Larson, Dan Massey, and Scott Rose. 2005. Protocol Modifications for the DNS Security Extensions. RFC 4035, IETF. doi:10.17487/RFC4035

[5] Roy Arends and Jakob Schlyter. [n. d.]. fpdns: DNS Server Fingerprinting Tool. DNS-OARC, https://github.com/kirei/fpdns. Accessed 2026.

[6] R. Can Aygun, Yehuda Afek, Anat Bremler-Barr, and Leonard Kleinrock. 2025. LAPRAD: LLM-Assisted Protocol Attack Discovery. In *IFIP Networking Conference (IFIP Networking)*. arXiv:2510.19264.

[7] Benjamin Beurdouche, Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cédric Fournet, Markulf Kohlweiss, Alfredo Pironti, Pierre-Yves Strub, and Jean Karim Zinzindohoue. 2015. A Messy State of the Union: Taming the Composite State Machines of TLS. In *2015 IEEE Symposium on Security and Privacy (SP)*. 535–552. doi:10.1109/SP.2015.39

[8] Karthikeyan Bhargavan, Christina Brzuska, Cédric Fournet, Matthew Green, Markulf Kohlweiss, and Santiago Zanella-Béguelin. 2016. Downgrade Resilience in Key-Exchange Protocols. In *2016 IEEE Symposium on Security and Privacy (SP)*. 506–525. doi:10.1109/SP.2016.37

[9] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32. doi:10.1023/A:1010933404324

[10] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees*. Wadsworth.

[11] Tim Bruijnzeels, Oleg Muravskiy, B. Weber, and Rob Austein. 2017. The RPKI Repository Delta Protocol (RRDP). RFC 8182, IETF. doi:10.17487/RFC8182

[12] Justin Cappos, Trishank Karthik Kuppusamy, Joshua Lock, Marina Moore, and Lukas Pühringer. 2026. The Update Framework Specification, Version 1.0.37. https://theupdateframework.github.io/specification/latest/.

[13] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research* 16 (2002), 321–357.

[14] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 785–794. doi:10.1145/2939672.2939785

[15] Davide Chicco and Giuseppe Jurman. 2020. The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation. *BMC Genomics* 21, 1 (2020), 6. doi:10.1186/s12864-019-6413-7

[16] Taejoong Chung, Roland van Rijswijk-Deij, Balakrishnan Chandrasekaran, David Choffnes, Dave Levin, Bruce M. Maggs, Alan Mislove, and Christo Wilson. 2017. A Longitudinal, End-to-End View of the DNSSEC Ecosystem. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, 1307–1322.

[17] Carlo Contavalli, Wilmer van der Gaast, David C. Lawrence, and Warren Kumari. 2016. Client Subnet in DNS Queries. RFC 7871, IETF. doi:10.17487/RFC7871

[18] Dave Crocker, Tony Hansen, and Murray S. Kucherawy. 2011. DomainKeys Identified Mail (DKIM) Signatures. RFC 6376 (STD 76), IETF. doi:10.17487/RFC6376

[19] Steve Crocker and Scott Rose. 2013. Signaling Cryptographic Algorithm Understanding in DNS Security Extensions (DNSSEC). RFC 6975, IETF. doi:10.17487/RFC6975

[20] Joao Damas, Michael Graff, and Paul Vixie. 2013. Extension Mechanisms for DNS (EDNS(0)). RFC 6891 (STD 75), IETF. doi:10.17487/RFC6891

[21] Donald E. Eastlake and Mark P. Andrews. 2016. Domain Name System (DNS) Cookies. RFC 7873, IETF. doi:10.17487/RFC7873

[22] Roy T. Fielding, Mark Nottingham, and Julian Reschke. 2022. HTTP Semantics. RFC 9110 (STD 97), IETF. doi:10.17487/RFC9110

[23] Kazunori Fujiwara, Akira Kato, and Warren Kumari. 2017. Aggressive Use of DNSSEC-Validated Cache. RFC 8198, IETF. doi:10.17487/RFC8198

[24] Roque Gagliano, Stephen Kent, and Sean Turner. 2013. Algorithm Agility Procedure for the Resource Public Key Infrastructure (RPKI). RFC 6916 (BCP 182), IETF. doi:10.17487/RFC6916

[25] Jason Goertzen and Douglas Stebila. 2023. Post-Quantum Signatures in DNSSEC via Request-Based Fragmentation. In *Post-Quantum Cryptography (PQCrypto) (LNCS, Vol. 14154)*. 535–564. arXiv:2211.14196. doi:10.1007/978-3-031-40003-2_20

[26] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Mądry, Bo Li, and Tom Goldstein. 2023. Dataset Security for Machine Learning: Data Poisoning, Backdoor Attacks, and Defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 2 (2023), 1563–1580. doi:10.1109/TPAMI.2022.3162397

[27] Xueluan Gong, Qian Wang, Yanjiao Chen, Wang Yang, and Xinchang Jiang. 2020. Model Extraction Attacks and Defenses on Cloud-Based Machine Learning Models. *IEEE Communications Magazine* 58, 12 (2020), 83–89. doi:10.1109/MCOM.001.2000196

[28] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. 2022. Why Do Tree-Based Models Still Outperform Deep Learning on Tabular Data?. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[29] Shuang Hao, Nick Feamster, and Ramakant Pandrangi. 2013. Towards Passive DNS Software Fingerprinting. In *Proceedings of the 9th Asian Internet Engineering Conference (AINTEC)*. ACM. doi:10.1145/2534142.2534144

[30] Wes Hardaker and Warren Kumari. 2025. Deprecating the Use of SHA-1 in DNSSEC Signature Algorithms. RFC 9905, IETF. doi:10.17487/RFC9905

[31] Wes Hardaker and Warren Kumari. 2025. DNSSEC Cryptographic Algorithm Recommendation Update Process. RFC 9904, IETF. doi:10.17487/RFC9904

[32] Elias Heftrig, Jean-Pierre Seifert, Haya Shulman, Michael Waidner, and Nils Wisiol. 2021. Poster: Predictive Cipher-Suite Negotiation for Boosting Deployment of New Ciphers. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '21)*. ACM, 2423–2425. doi:10.1145/3460120.3485349

[33] Elias Heftrig, Haya Shulman, Niklas Vogel, and Michael Waidner. 2024. The Harder You Try, The Harder You Fail: The KeyTrap Denial-of-Service Algorithmic Complexity Attacks on DNSSEC. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. arXiv:2406.03133. doi:10.1145/3658644.3690196

[34] Elias Heftrig, Haya Shulman, and Michael Waidner. 2022. SERVFAIL: The Unintended Consequences of Algorithm Agility in DNSSEC. ACM CCS 2022 (poster); arXiv:2205.10608. doi:10.1145/3548606.3563517

[35] Elias Heftrig, Haya Shulman, and Michael Waidner. 2023. Downgrading DNSSEC: How to Exploit Crypto Agility for Hijacking Signed Zones. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, 7429–7444.

[36] Elias Heftrig, Haya Shulman, and Michael Waidner. 2023. Poster: Off-Path DNSSEC Downgrade Attacks. In *Proceedings of the ACM SIGCOMM 2023 Conference*. doi:10.1145/3603269.3610840

[37] Amir Herzberg and Haya Shulman. 2013. Fragmentation Considered Poisonous, or: One-Domain-to-Rule-Them-All.org. In *2013 IEEE Conference on Communications and Network Security (CNS)*. doi:10.1109/CNS.2013.6682711

[38] Amir Herzberg and Haya Shulman. 2015. Cipher-Suite Negotiation for DNSSEC: Hop-by-Hop or End-to-End? *IEEE Internet Computing* 19, 1 (2015), 80–84. doi:10.1109/MIC.2015.3

[39] Amir Herzberg, Haya Shulman, and Bruno Crispo. 2014. Less is More: Cipher-Suite Negotiation for DNSSEC. In *Proceedings of the 30th Annual Computer Security Applications Conference (ACSAC)*. ACM, 346–355. doi:10.1145/2664243.2664283

[40] Paul E. Hoffman. 2023. DNS Security Extensions (DNSSEC). RFC 9364 (BCP 237), IETF. doi:10.17487/RFC9364

[41] Russ Housley. 2015. Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms. RFC 7696 (BCP 201), IETF. doi:10.17487/RFC7696

[42] Geoff Huston. 2023. Measuring the Use of DNSSEC. APNIC Labs, https://stats.labs.apnic.net/dnssec. Ad-based user-population measurement; ~30% validating and rising; accessed 2026.

[43] IANA. 2024. DNSSEC Algorithm Numbers. https://www.iana.org/assignments/dns-sec-alg-numbers/. Accessed 2026.

[44] Juyoul Lee, Sanzida Hoque, Abdullah Aydeger, and Engin Zeydan. 2025. Quantum-Resistant Domain Name System: A Comprehensive System-Level Study. arXiv preprint arXiv:2506.19943.

[45] John Levine. 2018. A New Cryptographic Signature Method for DomainKeys Identified Mail (DKIM). RFC 8463, IETF. doi:10.17487/RFC8463

[46] Tushin Mallick, Ashish Kundu, and Ramana Kompella. 2026. Study of Post Quantum Status of Widely Used Protocols. arXiv preprint arXiv:2603.28728.

[47] Keyu Man, Zhiyun Qian, Zhongjie Wang, Xiaofeng Zheng, Youjun Huang, and Haixin Duan. 2020. DNS Cache Poisoning Attack Reloaded: Revolutions with Side Channels. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1337–1350. doi:10.1145/3372297.3417280

[48] Moritz Müller, Matthew Thomas, Duane Wessels, Wes Hardaker, Taejoong Chung, Willem Toorop, and Roland van Rijswijk-Deij. 2019. Roll, Roll, Roll Your Root: A Comprehensive Analysis of the First Ever DNSSEC Root KSK Rollover. In *Proceedings of the Internet Measurement Conference (IMC)*. ACM, 1–14. doi:10.1145/3355369.3355570

[49] Yevheniya Nosyk, Maciej Korczyński, and Andrzej Duda. 2024. Guardians of DNS Integrity: A Remote Method for Identifying DNSSEC Validators Across the Internet. arXiv preprint arXiv:2405.19851.

[50] Audrey Randall, Enze Liu, Gautam Akiwate, Ramakrishna Padmanabhan, Geoffrey M. Voelker, Stefan Savage, and Aaron Schulman. 2020. Trufflehunter: Cache Snooping Rare Domains at Large Public DNS Resolvers. In *Proceedings of the ACM Internet Measurement Conference (IMC)*. ACM, 50–64. doi:10.1145/3419394.3423640

[51] Aditya Singh Rawat and Mahabir Prasad Jhanwar. 2023. Post-Quantum DNSSEC over UDP via QNAME-Based Fragmentation. In *Security, Privacy, and Applied Cryptography Engineering (SPACE) (LNCS, Vol. 14412)*. 66–85. doi:10.1007/978-3-031-51583-5_4

[52] Eric Rescorla. 2018. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, IETF. doi:10.17487/RFC8446

[53] Scott Rose. 2025. *Secure Domain Name System (DNS) Deployment Guide*. Technical Report NIST SP 800-81r3 (initial public draft). National Institute of Standards and Technology. doi:10.6028/NIST.SP.800-81r3.ipd

[54] Amir Mahdi Sadeghzadeh, Saeed Shiravi, and Rasool Jalili. 2020. Adversarial Network Traffic: Towards Evaluating the Robustness of Deep Learning-Based Network Traffic Classification. arXiv preprint arXiv:2003.01261.

[55] Justin Samuel, Nick Mathewson, Justin Cappos, and Roger Dingledine. 2010. Survivable Key Compromise in Software Update Systems. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS)*. doi:10.1145/1866307.1866315

[56] Stefan Santesson, Michael Myers, Rich Ankney, Ambarish Malpani, Slava Galperin, and Carlisle Adams. 2013. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP. RFC 6960, IETF. doi:10.17487/RFC6960

[57] Jim Schaad, Blake Ramsdell, and Sean Turner. 2019. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification. RFC 8551, IETF. doi:10.17487/RFC8551

[58] Kyle Schomp, Tom Callahan, Michael Rabinovich, and Mark Allman. 2013. On Measuring the Client-Side DNS Infrastructure. In *Proceedings of the 2013 Internet Measurement Conference (IMC)*. ACM, 77–90. doi:10.1145/2504730.2504734

[59] Syed W. Shah, Lei Pan, Din Duc Nha Nguyen, Robin Doss, Warren Armstrong, and Praveen Gauravaram. 2024. Double-Signed Fragmented DNSSEC for Countering Quantum Threat. arXiv preprint arXiv:2411.07535.

[60] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy (SP)*. 3–18. doi:10.1109/SP.2017.41

[61] Kris Shrishak and Haya Shulman. 2021. Negotiating PQC for DNSSEC. In *51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-S)*. doi:10.1109/DSN-S52858.2021.00015

[62] Julio Gento Suela, Javier Blanco-Romero, Florina Almenares Mendoza, and Daniel Díaz-Sánchez. 2025. Implementing and Evaluating Post-Quantum DNSSEC in CoreDNS. In *28th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*. arXiv:2507.09301. doi:10.1109/MSWiM67937.2025.11308970

[63] Roland van Rijswijk-Deij, Taejoong Chung, David Choffnes, Alan Mislove, and Willem Toorop. 2017. The Root Canary: Monitoring and Measuring the DNSSEC Root Key Rollover. In *Proceedings of the SIGCOMM Posters and Demos*. doi:10.1145/3123878.3131987

[64] Roland van Rijswijk-Deij, Anna Sperotto, and Aiko Pras. 2014. DNSSEC and Its Potential for DDoS Attacks: A Comprehensive Measurement Study. In *Proceedings of the 2014 Internet Measurement Conference (IMC)*. ACM, 449–460. doi:10.1145/2663716.2663731

[65] Samuel Weiler and David Blacka. 2013. Clarifications and Implementation Notes for DNS Security (DNSSEC). RFC 6840, IETF. doi:10.17487/RFC6840

[66] Paul Wouters and Ondřej Sury. 2019. Algorithm Implementation Requirements and Usage Guidance for DNSSEC. RFC 8624, IETF. doi:10.17487/RFC8624

[67] Chengxi Xu, Yunyi Zhang, Fan Shi, Hong Shan, Bingyang Guo, Yuwei Li, and Pengfei Xue. 2023. Measuring the Centrality of DNS Infrastructure in the Wild. *Applied Sciences* 13, 9 (2023), 5739. doi:10.3390/app13095739

[68] Michał Zalewski. [n. d.]. p0f v3: Passive Traffic Fingerprinting. https://lcamtuf.coredump.cx/p0f3/. Accessed 2026.

[69] Qifan Zhang, Zilin Shen, Imtiaz Karim, Elisa Bertino, and Zhou Li. 2025. Proving DNSSEC Correctness: A Formal Approach to Secure Domain Name Resolution. arXiv preprint arXiv:2512.11431.