Domain Name System without Root Servers

Matthäus Wander, Christopher Boelmann, and Torben Weis

University of Duisburg-Essen
Duisburg, Germany

Abstract. We present a variation to the infrastructure of the Domain Name System (DNS) that works without DNS root servers. This allows to switch from a centralized trust model (root) to a decentralized trust model (top-level domains). By dropping DNS root in our approach, users have one entity less that they must trust. Besides trust issues, not relying on DNS root means that DNS root servers are no longer a central point of failure. Our approach is minimally invasive, builds on established DNS architecture and protocols and supports the DNS Security Extensions (DNSSEC). Furthermore, we designed our approach as an opt-in technology. Thus, each top-level domain operator can decide whether to support rootless DNS or not.
The challenge of a rootless DNS is to keep track of changing IP addresses of top-level domain servers and to handle key rollovers, which are part of normal DNSSEC operation. Top-level domains opting in to rootless DNS must follow constraints regarding the frequency of changes of IP addresses and DNSSEC keys. We conducted a four-year measurement to show that 82% respectively 72% of top-level domains fulfill these constraints already.

Keywords: Domain Name System, DNSSEC, Infrastructure security

1 Introduction

The Domain Name System (DNS) is a critical infrastructure for the whole Internet. In order to resolve a domain name, clients iterate through the hierarchical namespace from root to the leaf. The availability of the DNS thus depends on a reliable operation of the root. With the DNS Security Extensions (DNSSEC), the root also serves as trust anchor for authorizing cryptographic keys that are used for signing domain name entries. The security of DNSSEC thus depends on a trusted and proper root key management.

In this paper we explore the design of a rootless Domain Name System. The objective of this system is to have independent top-level domains (TLDs), which do not depend on a root authority. The trust is distributed to coequal TLD operators, whose control is limited to their respective namespace. We argue that the dependency on root can be eliminated with little technical changes for most TLDs and without introducing new attack vectors. The two components of this approach—*priming* and *trust anchor updates*—have been originally designed to work on root level, but we show that they are also applicable for TLDs.

The proposed approach is minimally invasive and reuses the existing network protocol, methods and infrastructure. Unlike peer-to-peer-based approaches, this allows for an incremental deployment and shares the performance and usability characteristics of the proven-in-practice DNS.

2 Motivation

Trust. This work is motivated by removing the trust dependency upon root without lessening the security guarantees of DNSSEC. The DNS root has the technical authority to answer any domain or to delegate any domain to another organization. The root key is configured as trust anchor in validating DNSSEC clients. A compromised root key thus allows an attacker to fabricate malicious responses with a valid DNSSEC signature, e.g. as part of a man-in-the-middle attack. Apart from the threat of key theft, the organizations handling the root key are entrusted not to misuse the key for actions that are not in accordance with objectives of the global DNS community. This includes not to use the root authority as a political instrument for influencing Internet governance. Name resolution without root removes this burden of trust in influential institutions.

Reliability. Another factor is reliability, as the availability of DNS depends on the availability of the root. Techniques like caching and anycast [1] are used to reduce the load on root or add redundancy. Though this cushions the impact of malfunctions or denial of service attacks, the threat remains in principle. A rootless name resolution can serve as backup mechanism to ensure DNS service availability in case of severe failures of the root.

Use case. One of the possible use cases of a rootless DNS are redundant domain names. For example, consider the name "`www.example.br+pl+cz`" as three distinct names `www.example.br`, `www.example.pl` and `www.example.cz` that should all evaluate to the same IP address. Whenever a client resolves a redundant name, e.g. in a weblink or in the configuration of an email client, the client performs a majority voting over all three domains. With rootless DNS and redundant domain names, there is no single entity that has the power to misdirect the name resolution.

3 Background

3.1 DNS and DNSSEC

The DNS architecture consists of resolvers (clients) and name servers. The hierarchical namespace is structured as a tree, which is cut into non-overlapping zones. A *zone* is a collection of domain names below a particular part of the namespace, e.g. `com` or `example.net`. Zones contain either the actual data entities or delegate subparts of the namespace to other name servers. The root of the DNS tree comprises the root zone, which delegates TLDs like `com` or `uk` to the corresponding TLD name servers. A *delegation* consists of a set of authoritative name servers for the child zone and their IP addresses.

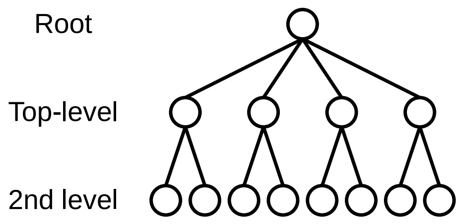

Fig. 1. DNSSEC trust model.

The DNS Security Extensions (DNSSEC) introduce public-key cryptography into that system. The trust model builds on top of the hierarchical namespace, as shown in Fig. 1. Domain name records are signed with the private key of the zone administrator and resolvers verify the signature after the corresponding public key has been authenticated. Delegations additionally contain the fingerprint (hash value) of the public key of the child zone. Thus, resolvers authenticate zone keys by following the chain of keys up to the root, which is signed by a key known to the resolver. Such a well-known key serves as *trust anchor*.

3.2 Root Zone Management

Several entities are involved in the management and operation of the root zone. TLD operators request changes of TLD delegations from the *Internet Corporation for Assigned Names and Numbers* (ICANN) respectively its close affiliate *Public Technical Identifiers* (PTI), who manages the TLD contacts and delegations [2]. ICANN/PTI vets change requests for technical and formal correctness and forwards them to Verisign, who implements the change in the root zone and distributes it to the operators of the 13 root name servers. ICANN is a U.S.-based non-profit organization and Verisign is a U.S.-based corporation. 7 root name servers are operated by U.S.-based organizations, 3 are operated by U.S. governmental or military agencies, and the operators of the remaining 3 servers are located in Sweden, the Netherlands and Japan, respectively. Historically, the U.S. government oversaw the root zone operation, but stepped back from this role in October 2016.

There are two root DNSSEC keys: the root *Key Signing Key* (KSK) is configured as trust anchor on DNSSEC validators worldwide. ICANN stores the private part of the root KSK at two redundant colocation facilities and uses it for authorization of the root *Zone Signing Key* (ZSK), which is replaced regularly in the root zone. Verisign owns the private part of the root ZSK and uses it for signing the root zone contents, which have been provided and approved by ICANN respectively its affiliate PTI. Both the root KSK and ZSK are technically eligible for signing and authenticating any domain or any TLD delegation.

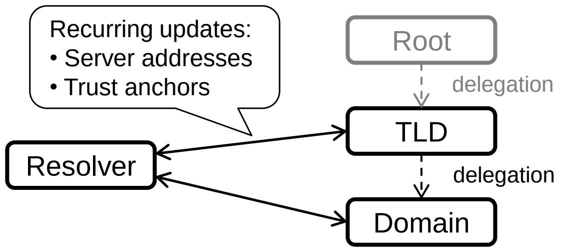

Fig. 2. Name resolution without asking root. Resolvers query TLD servers regularly for the current TLD server set (*priming*) and the current TLD key set (*trust anchor update*).

4 Approach

The name resolution that we intend in our approach is shown in Fig. 2: resolvers skip the root and start name resolution at top level, which is one level below root. The resolvers must hold the information that would be otherwise served by root: the names of the TLDs, the set of authoritative name servers and their IP addresses, and the copy or a secure fingerprint of the TLD public key. The challenge of our approach is for resolvers to keep this information up-to-date, as TLD delegations occasionally change. We show that it is possible to update the TLD server addresses (Section 4.2) and trust anchors (Section 4.3) without asking or trusting root.

4.1 Bootstrapping

Bootstrapping is the process of initializing a rootless resolver with the TLD delegation data required for resolving domain names. The secure retrieval of trust anchors is sensitive in particular because a successful attack at this point compromises the security of DNSSEC. The bootstrapping must be performed once at the installation time of a DNS resolver. At this point the user or system administrator relies on their operating system vendor or a DNS software vendor to retrieve the resolver software. This is a trusted path to ensure that the behavior of the resolver is not manipulated, otherwise this would nullify any security mechanisms like DNSSEC validation. In present-day DNS, the resolver software is shipped with the IP addresses of the root name servers and the root KSK as trust anchor. Our approach complements this existing bootstrapping channel with a list of TLDs, each with an initial set of TLD servers and a TLD trust anchor. The system can bootstrap from aged data within certain limitations (cf. Section 4.4), e.g. from a USB stick or another physical storage medium obtained from a software vendor.

An alternative bootstrapping channel is by manual intervention, i.e. a person inspects and copies the TLD trust anchor and server addresses from a source deemed to be trusted. TLD operators should foster the dissemination of their

TLD delegation data over several publication channels, e.g. on their website, by sending changes via email or snail mail to subscribers, or by offering voice message via telephone. Though none of these invididual methods is inherently secure, each of them contributes to a human decision whether the data is sufficiently trusted. Manual bootstrapping is cumbersome and we expect that the general user base will not use it, though individual operators of high-value resolvers e.g. in industry or government networks have the possibility to do so. The target audience of manual bootstrapping are vendors of DNS software and network appliances, who redistribute the TLD data over their automated software update mechanisms.

4.2 Priming

Resolvers perform priming to initialize an empty cache after startup with information about which server to ask first during name resolution. In present-day DNS, resolvers send priming queries to one of the root name servers [3], with the IP addresses given in a configuration file `root.hints` or built into the resolver software. The query asks for the set of authoritative name servers for the root domain ("`. IN NS`"). The response contains the root server names together with their IPv4 and IPv6 addresses (so called *glue records*). In our approach, resolvers skip root and ask each TLD for their authoritative name servers ("`tld IN NS`"). The TLD server addresses have been initially retrieved during bootstrapping. When the priming query times out due to an IP address change in the meantime, the resolver retries with another known TLD server address. The priming response will be cached for the time interval indicated by the Time-to-Live (TTL) field. Subsequent queries for another name under the same TLD will be thus served without priming. When different TLDs are being queried, each TLD will be primed once and remain in cache until the TTL times out or the resolver shuts down.

The priming response contains the set of name servers that are currently authoritative for the TLD, together with the glue records of these servers. In our approach, priming responses are supposed to be signed with DNSSEC and are validated with the trust anchor for the respective TLD, which has been retrieved during bootstrapping. Though the server names are signed, the IP addresses are not because DNSSEC does not sign glue records by design. Our approach does not intend to change the DNSSEC signing or validation semantics; instead, resolvers emit follow-up resolution attempts for each server name to one of the TLD servers. Thus, resolvers will validate all contents of the priming response, including the authenticity of the name-to-IP-address mapping at the cost of additional network queries (one per each server name).

Once the priming response has been received and all server names have been resolved and validated, resolvers store the new server addresses locally for later access. Whenever the TLD operator adds or replaces a server IP address, the change will be distributed to all resolvers with the next priming query and response. Thereby the TLD server addresses, which may change over time, are kept up-to-date with an in-band update mechanism.

4.3 Updating Trust Anchors

In the rootless approach, resolvers use locally stored TLD public keys as trust anchors. DNSSEC-signed responses are validated by authenticating the chain of keys from some leaf domain `www.example.tld` up to a TLD trust anchor. In normal operation, trust anchors and other keys must be replaced periodically to avoid that a key is broken eventually. The key replacement procedure is called *key rollover* and does not replace keys at an instant of time, but gradually introduces a new key and then withdraws the old key. RFC 5011 [4] specifies an automated update mechanism for trust anchors, which we utilize to keep the TLD trust anchors up-to-date. Resolvers regularly retrieve the set of all TLD keys by querying for "`tld IN DNSKEY`". The TLD operator may introduce new keys into the set, as long as the set is signed by the established TLD trust anchor. The new key cannot be used for signing zone data for a hold-down time of 30 days while resolvers learn of the new key soon to be used. The resolver will thus store two redundant trust anchors for a TLD, both authorized for signing operations. In order to remove a deprecated trust anchor, the TLD operator sets a revocation flag in the key set to indicate which key is ought to be deleted. Resolvers check the integrity and authenticity of the revocation flag during DNSSEC validation.

4.4 Opt-In and Commitment

The rootless name resolution is designed as opt-in service for both, TLD operators and resolver operators. Our motivation for an optional service is as follows:

- It allows gradual and parallel deployment in the existing DNSSEC ecosystem.
- TLD operators that opt-in are making a commitment to serve stable parameters over an extended period of time. If they cannot or do not want to commit, they should not opt-in.
- Resolver operators decide whether the rootless approach is useful to them, e.g. subject to a trust assessment in root.

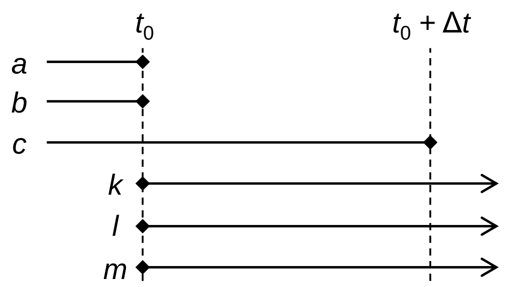

Fig. 3. IP address update scheme for TLDs.

TLD operators indicate support by setting a flag in a special-purpose DNS record in the TLD zone, e.g. by reutilizing the TXT record or by specifying a dedicated AUTONOMOUS record type. Furthermore, TLD operators cooperate with DNS software vendors to publish their bootstrapping data. When the opt-in flag is set, the TLD operator commits to operate long-lived name servers and trust anchors. The commitment period Δt is configurable per TLD in the above-mentioned record. The commitment applies to server IP addresses and trust anchors so that resolvers will know when to refresh a TLD before potential expiration.

When a TLD operators intends to replace all server **IP addresses**, they must continue to operate at least one server on an old IP address for at least Δt. However, with n server addresses, up to $n - 1$ addresses can be removed or replaced without any special considerations. An example is shown in Fig. 3: addresses a and b are removed, while address c is held back for Δt. Thus the priming succeeds and resolvers will learn of newly introduced server IP addresses k, l and m, which they can subsequently use.

TLD operators must retain a **trust anchors** for signing for at least Δt (though different Δt values can be used for IP addresses and trust anchors). Unlike server IP addresses, there is typically only one established trust anchor for a domain, but otherwise the same considerations as above apply. When replacing trust anchors periodically, there must be an overlap of Δt of the old and new key so that resolvers will authenticate newly introduced keys with the trust anchor previously known to them.

We can formalize the commitment period as

$$\exists x \in RESP : active(x) > \Delta t, \tag{1}$$

where $RESP$ is the set of parameters publicly released in priming or trust anchor responses, $active(x)$ is the lifetime of a parameter x (IP address or trust anchor) before changing or removing it. The rootless approach will be self-sustaining if resolvers manage to run the priming and trust anchor update every $\Delta u \leq \Delta t$. Our recommendation is to specify an ample commitment period of $\Delta t = 1$ year to account for extended offline times of resolvers running on end-user devices. Most TLDs already fulfill this commitment as we show in the feasibility study in Section 6. If a resolver is offline for $> \Delta t$, it may miss an update and will need to bootstrap the TLD data from scratch.

5 Security Discussion

Trust. By cutting off root, we have eliminated a single point of trust. The trust model is still a hierarchical one, separately for each TLD. Each TLD is a point of trust on its own, because each operator is able to tamper with their registered domains. However, this has been true before, thus our approach does not create any new attack vector. The authority of a TLD operator is limited to their own namespace, whereas the authority of root includes all TLDs. Domain registrants can choose between several operators independent of each other, which are based in different countries and jurisdictions. This allows for decentralized domain setups, e.g. redundant domains under different TLDs.

TLD or resolver operators do not need to cooperate with root to deploy the rootless approach. A trusted bootstrapping channel is required to securely initialize a TLD, which constitutes a weak point of our approach. We suggest to involve the operating system or DNS software vendor for this purpose because we can reuse the existing trust relationship and communication channel that is used for distributing software security updates. Again, this does not introduce a new attack vector in addition to potential attacks that are already possible. Once bootstrapped, the system will update in-band via DNSSEC-secured communication with TLD servers. The approach is designed as opt-in service, which means that resolvers will continue to rely on root for those TLDs that have not opted-in, but the influence of root is confined to those TLDs.

Key rollovers. The commitment period Δt limits the frequency of trust anchor rollovers for TLDs. For those TLDs, which roll the key more often, this imposes a security degradation. With appropriately sized keys, a rollover every one or two years is sufficiently secure. Unscheduled emergency rollovers, though, will lead to a service degradation and resolvers will have to bootstrap that TLD again. Unscheduled key rollovers are exceptional events that should not occur on a regular basis. However, they potentially degrade the availability compared to using the root as backup authority.

Privacy. As a side effect, omitting root accounts for one fewer authority that is able to inspect query contents. This slightly improves the privacy of clients, albeit to a small degree, because queries are still transmitted in cleartext to TLDs. The effect is not identical but roughly comparable to DNS query name minimization [5], which also attempts to hide the query name from root, amongst others.

6 Feasibility Study

In this section we provide a feasibility study for our approach to a rootless DNS by analyzing the root zone development over a time period of 4 years from April 2013 to May 2017. We collected data almost daily by downloading the root zone from ICANN servers and querying TLD servers directly to gather the TLD Key Signing Keys (KSK), which are our trust anchors. The measurement period overlaps with ICANN's introduction of new generic TLDs [6]; out of 1549 TLDs in our data, the majority has been introduced since October 2013. We discard TLDs that appeared for less than one year in our data because we require longer periods of time for evaluation of long-term behavior. This yields a data set of 1317 TLDs for this study, composed of 306 old and 1011 new TLDs.

6.1 Frequency of IP Address Changes

First we determine the period after when a TLD has replaced all of their server IP addresses. This enables us to approximate a threshold of how long a resolver may

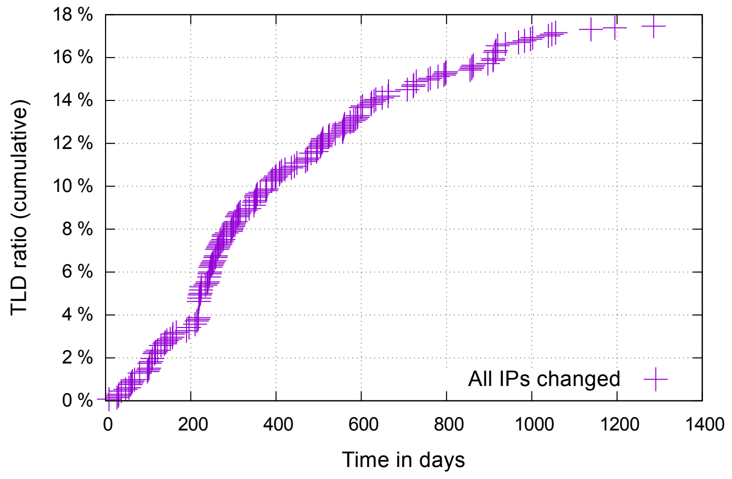

Fig. 4. Duration after when TLDs replaced all IP addresses ($N = 1317$).

be offline until it will be unable to find the servers of a TLD using priming when getting back online. If the resolver looses track of the current server IP addresses, it will need to bootstrap again with the mechanism explained in Section 4.1.

Fig. 4 shows the cumulated amount of TLDs, whose set of IPv4 and IPv6 addresses had completely changed after a period shown on the x-axis. Altogether 233 TLDs (17.7%) changed their server IP addresses completely during the measurement. Most of these were new TLDs, whose operations may not have matured yet; 26 old TLDs (2%) changed their whole server address set. For the other 1084 TLDs (82.3%) at least one IP address lasted for the whole measurement duration. The majority of TLDs would have thus been reachable with rootless DNS even without any recurring priming queries during our measurement period.

Considering the lifetime of IP addresses, 943 TLDs (71.6%) kept *each* of their addresses for ≥ 1 year. Thus, the majority overfulfills our requirement of committing to at least *one* IP address for $\Delta t \geq 1$ year.

Whether a TLD becomes unavailable for a rootless resolver depends on its update interval Δu. The update interval limits the maximum tolerable offline time of a resolver, since the device must be powered on and connected to the Internet to perform the update. We now simulate different intervals $\Delta u \in \{365, 90, 14, 7, 1\}$ (in days) for the periodic priming queries. The simulation is driven with our real-world measurement data and checks all possible offsets, e.g. for $\Delta u = 14$ days we conduct 14 simulations starting on day 1, 2, ..., 14. We consider the worst-case result, i.e. a TLD is unavailable if at least one of simulation offsets looses track of the server IP addresses.

Fig. 5 shows the result subject to different Δu. Running priming once in year slightly improves the availability over running no updates at all, but 221 TLDs

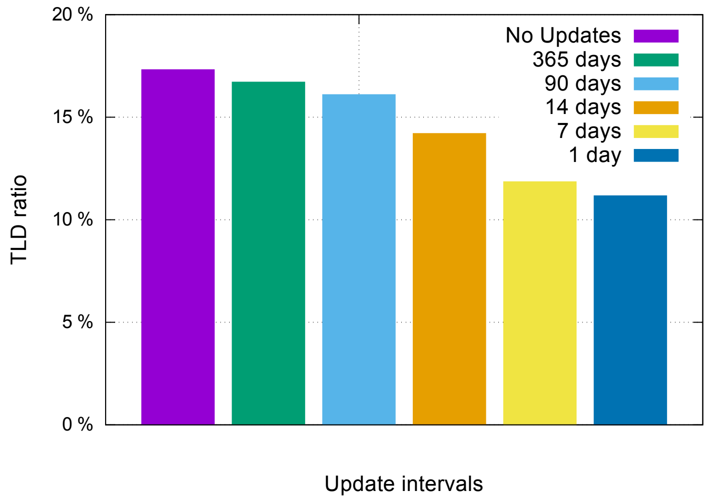

Fig. 5. Ratio of unavailable TLDs subject to resolver update intervals.

(16.8%) become unavailable eventually. Reducing the update interval steadily decreases the amount of TLDs that the resolver would have lost track of. However, even with daily priming 147 TLDs (11.2%) would have become unavailable because they replaced all server IP addresses at once. Note that these are conservative estimations because our data does not cover whether the TLD operator continued to serve the old IP addresses after switching to new addresses. It is good operational practice to maintain servers on both IP address sets for a transition period (e.g. used by the D-root server operator [7]) and this might actually mitigate an outage of our rootless approach.

6.2 Frequency of Trust Anchors Rollover

In the following study we analyze how often TLD operators replaced the trust anchor (i.e. the public *Key Signing Key* or KSK) within our 4-year observation. Note that not all TLDs are already signed with DNSSEC: we consider 1174 TLDs (89.1%) in this study that were signed for at least part of our observation period.

Fig. 6 shows the average lifetime of a KSK before being replaced. The first thing to notice is that 351 TLDs (29.9%) never replaced the KSK during our observation period. An additional 488 TLDs (41.6%) replaced the KSK, but less often than once a year, i.e. maintained one key for more than one year on average. Thus, 839 TLDs (71.5%) satify our proposed commitment period Δt in our observation on average.

A minority of TLDs rolls their keys very frequently: 175 TLDs (14.9%) replaced their KSK every 90 days or less. These operators would have to roll their KSK less often if they chose to opt-in. Instead, the operators could increase the KSK bit length to compensate for long-lived keys.

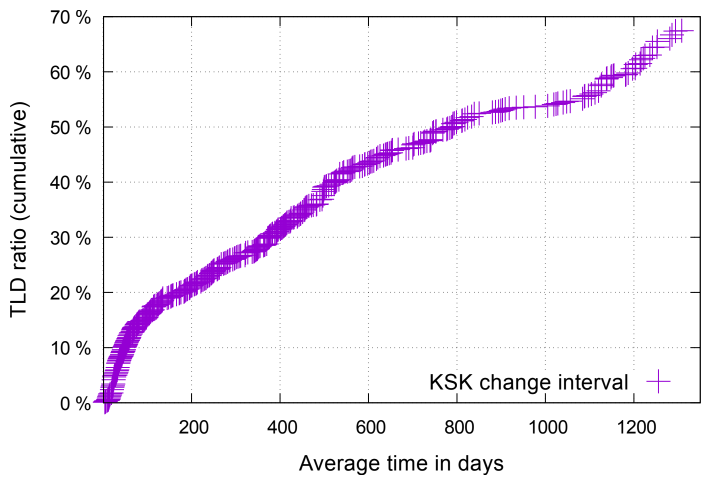

Fig. 6. KSK change interval of TLDs ($N = 1174$).

6.3 Efficiency

The rootless approach requires periodic retrieval of TLD server IP addresses and trust anchors. However, regular domain name lookups below a TLD require these queries anyway. When a TLD is in active use, the message overhead for the rootless approach will be insignificant compared to regular DNS operations. We leave it to future studies to validate this claim with a quantification.

The message size of TLD trust anchors increases because the rootless approach requires a continous overlap during the commitment period when intending to replace keys regularly. This disadvantage can be mitigated by using elliptic-curve signatures, which are considerably shorter than RSA signatures with a similar level of security [8].

7 Related Work

Alternative DNS roots have emerged in the past, which were operated independently from the ICANN-coordinated root. Mueller [9] explained in 2002 that competing DNS roots were caused by a disjunction between the demand for and supply of new TLDs. ICANN introduced 15 new generic TLDs in 2000, 2004 and 2011 and is in the process of introducing more than thousand new TLDs [6] since 2013. The Open Root Server Network (ORSN) [10] is an alternative DNS root that was created explicitly for strategical reasons only and without serving any other data than the ICANN root. The objective of ORSN is to create a counterpart, independent of any influence from the ICANN root. Our approach differs from the concept of competing roots by eliminating the need for a central root authority.

Peer-to-peer-based naming systems have been proposed to address the centralization of the Domain Name System, e.g. CoDoNS [11], Namecoin, GNS [12] and others [13–15]. These systems are entirely different than the DNS, allowing to take interesting technological approaches to improve certain aspects, but are weaker in other aspects than a server-based system. Peer-to-peer-based systems require a decentralized overlay network, e.g. a distributed hash table (DHT), to achieve its function cooperatively without a central instance. Typical weak spots of DHT-based systems are higher lookup latency, loss of locality and loss of resistence against sybil attacks. Our solution in this paper is a small improvement but one that can be gradually deployed in the well-established DNS, sharing the existing namespace, infrastructure and performance characteristics. While we omit the central root authority, we do not achieve the same degree of decentralization like peer-to-peer-based systems, because each TLD operator is still an authority for their part of the domain namespace. Thus, our approach is a compromise between the degree of decentralization/control achieved by peer-to-peer-based systems and performance/stability of the existing Domain Name System.

Massey et al. [16] identified that the tree-based DNSSEC trust model is detrimental due to a single point of failure and undesirable trust relationships. They discussed the web of trust and mesh of trust approaches, which allow for trust relationships that do not follow the hierarchical namespace.

Malone [17] suggested to distribute a copy of the root zone to resolvers for improving performance and scalability. There is no trust improvement because the root zone is copied from the root authority without changes.

Kuerbis and Mueller [18] proposed to distribute the root signing authority to different actors to increase transparency and eliminate the threat of political interference. ICANN is using a centralized root signing procedure instead, but they invite community representatives to participate as independent bystanders or recovery key holders [2] for the sake of transparency. With our rootless approach, the power of the root authority is inherently distributed to independent TLD operators.

8 Conclusions

Our analysis has shown that a rootless DNS/DNSSEC can be implemented with minimal changes to the existing infrastructure. Already today, most TLD operators are handling IP address updates (82%) and key rollovers (72%) in sufficiently large intervals. Those TLD operators who change IP addresses or keys more frequently could continue to do so, because our approach is entirely opt-in. Both, TLD providers and client-side resolver operators can decide whether or not to use the rootless name resolution. This allows a smooth phase-in of our proposed approach and coexistence in the same DNS ecosystem without having to switch to another naming system.

The benefit of rootless DNS is that we moved from a centralized to a decentralized trust model. That means there is no longer a single entity controlling the entire namespace. The primary stakeholders are providers and users of country-code TLDs, who are able to operate a stable naming infrastructure autonomously without potential interference from root. Though we are focusing on top-level providers, the rootless approach could be used on lower levels in the DNS hierarchy as well.

In a next step, we want to evaluate how such a system performs when the DNS resolvers are located on end-user devices. As of today, clients typically rely on their network operator to resolve and validate domain names. Name resolution on user devices allows end-to-end security with DNSSEC, for example when authenticating digital certificates via *DNS-Based Authentication of Named Entities* (DANE). However, this increases the load on TLD servers since each client has to run the update procedure as described in this paper. It would be interesting to quantify the efficiency of the rootless approach with practical evaluations.

References

1. J. Abley and K. Lindqvist, "Operation of Anycast Services," RFC 4786 (Best Current Practice), Internet Engineering Task Force, Dec. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4786.txt
2. Root Zone KSK Policy Management Authority, "DNSSEC Practice Statement for the Root Zone KSK Operator," Oct. 2016. [Online]. Available: https://www.iana.org/dnssec/icann-dps.txt
3. P. Koch, M. Larson, and P. Hoffman, "Initializing a DNS Resolver with Priming Queries," RFC 8109, Mar. 2017. [Online]. Available: http://www.ietf.org/rfc/rfc8109.txt
4. M. St.Johns, "Automated Updates of DNS Security (DNSSEC) Trust Anchors," RFC 5011, Internet Engineering Task Force, September 2007. [Online]. Available: http://www.ietf.org/rfc/rfc5011.txt
5. S. Bortzmeyer, "DNS Query Name Minimisation to Improve Privacy," RFC 7816 (Experimental), Internet Engineering Task Force, Mar. 2016. [Online]. Available: http://www.ietf.org/rfc/rfc7816.txt
6. Internet Corporation For Assigned Names and Numbers. New Generic Top-Level Domains. [Online]. Available: https://newgtlds.icann.org
7. M. Lentz, D. Levin, J. Castonguay, N. Spring, and B. Bhattacharjee, "D-mystifying the D-root Address Change," in *Proceedings of the 2013 Conference on Internet Measurement Conference*, ser. IMC '13. New York, NY, USA: ACM, 2013, pp. 57–62.
8. R. van Rijswijk-Deij, A. Sperotto, and A. Pras, "Making the Case for Elliptic Curves in DNSSEC," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 13–19, Sep. 2015. [Online]. Available: http://doi.acm.org/10.1145/2831347.2831350
9. M. L. Mueller, "Competing dns roots: creative destruction or just plain destruction," *Journal of Network Industries*, vol. 3, p. 313, 2002.
10. Open Root Server Network. [Online]. Available: http://www.orsn.org
11. V. Ramasubramanian and E. G. Sirer, "The Design and Implementation of a Next Generation Name Service for the Internet," in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4. ACM, 2004, pp. 331–342.

12. M. Wachs, M. Schanzenbach, and C. Grothoff, "A Censorship-Resistant, Privacy-Enhancing and Fully Decentralized Name System," in *Cryptology and Network Security*, ser. Lecture Notes in Computer Science, D. Gritzalis, A. Kiayias, and I. Askoxylakis, Eds. Springer International Publishing, 2014, vol. 8813, pp. 127–142.
13. R. Cox, A. Muthitacharoen, and R. Morris, "Serving dns using a peer-to-peer lookup service," in *Peer-to-Peer Systems*, ser. Lecture Notes in Computer Science, P. Druschel, F. Kaashoek, and A. Rowstron, Eds. Springer Berlin Heidelberg, 2002, vol. 2429, pp. 155–165.
14. M. Theimer and M. Jones, "Overlook: Scalable Name Service on an Overlay Network," in *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, 2002, pp. 52–61.
15. P. Danielis, V. Altmann, J. Skodzik, T. Wegner, A. Koerner, and D. Timmermann, "P-DONAS: A P2P-Based Domain Name System in Access Networks," *ACM Trans. Internet Technol.*, vol. 15, no. 3, pp. 11:1–11:21, Sep. 2015. [Online]. Available: http://doi.acm.org/10.1145/2808229
16. D. Massey, E. Lewis, O. Gudmundsson, R. Mundy, and A. Mankin, "Public key validation for the DNS security extensions," in *DARPA Information Survivability Conference &amp; Exposition II, 2001. DISCEX'01. Proceedings*, vol. 1. IEEE, 2001, pp. 227–238.
17. D. Malone, "The root of the matter: Hints or slaves," in *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '04. New York, NY, USA: ACM, 2004, pp. 15–20.
18. B. Kuerbis and M. Mueller, "Securing the Root: A Proposal for Distributing Signing Authority," *Paper IGP07-002*, 2007.