DNS Operations
Internet-Draft
Intended status: Standards Track
Expires: April 22, 2019

T. Finch
University of Cambridge
E. Hunt
ISC
P. van Dijk
PowerDNS
A. Eden
DNSimple

October 19, 2018

Address-specific DNS aliases (ANAME) draft-ietf-dnsop-aname-02

Abstract

This document defines the "ANAME" DNS RR type, to provide similar functionality to CNAME, but only for type A and AAAA queries. Unlike CNAME, an ANAME can coexist with other record types. The ANAME RR allows zone owners to make an apex domain name into an alias in a standards compliant manner.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 22, 2019.

Table of Contents

1. Introduction 2
1.1. Overview 3
1.2. Terminology 4
2. The ANAME resource record 5
2.1. Presentation and wire format 5
2.2. Coexistence with other types 5
3. Additional section processing 5
3.1. Address queries 6
3.2. ANAME queries 6
4. Substituting ANAME sibling address records 6
5. ANAME processing by primary masters 7
5.1. Implications 8
5.2. Alternatives 8
6. ANAME processing by resolvers 9
7. IANA considerations 10
8. Security considerations 10
9. References 11
9.1. Normative References 11
9.2. Informative References 11
Appendix A. Acknowledgments 12
Appendix B. Implementation status 12
Appendix C. Historical note 12
Appendix D. On preserving TTLs 13
D.1. Query bunching 14
D.2. Upstream caches 14
D.3. ANAME chains 15
D.4. TTLs and zone transfers 15
Appendix E. Answer vs Additional sections 15
Appendix F. Changes since the last revision 16
Authors' Addresses 16

1. Introduction

It can be desirable to provide web sites (and other services) at a bare domain name (such as "example.com") as well as a service-specific subdomain ("www.example.com").

If the web site is hosted by a third-party provider, the ideal way to provision its name in the DNS is using a CNAME record, so that the third party provider retains control over the mapping from names to

IP address(es). It is now common for name-to-address mappings to be highly dynamic, dependent on client location, server load, etc.

However, CNAME records cannot coexist with other records. (The reason why is explored in Appendix C). This means they cannot appear at a zone apex (such as "example.com") because of the SOA, NS, and other records that have to be present there. CNAME records can also conflict at subdomains, for example if "department.example.edu" has separately hosted mail and web servers.

Redirecting website lookups to an alternate domain name via SRV or URI resource records would be an effective solution from the DNS point of view, but to date this approach has not been accepted by browser implementations.

As a result, the only widely supported and standards-compliant way to publish a web site at a bare domain is to place A and/or AAAA records at the zone apex. The flexibility afforded by CNAME is not available.

This document specifies a new RR type "ANAME", which provides similar functionality to CNAME, but only for address queries (i.e., for type A or AAAA). The basic idea is that the address records next to an ANAME record are automatically copied from and kept in sync with the ANAME target's address records. The ANAME record can be present at any DNS node, and can coexist with most other RR types, enabling it to be present at a zone apex, or any other name where the presence of other records prevents the use of CNAME.

Similar authoritative functionality has been implemented and deployed by a number of DNS software vendors and service providers, using names such as ALIAS, ANAME, apex CNAME, CNAME flattening, and top level redirection. These mechanisms are proprietary, which hinders the ability of zone owners to have the same data served from multiple providers, or to move from one provider to another. None of these proprietary implementations includes a mechanism for resolvers to follow the redirection chain themselves.

1.1. Overview

The core functionality of this mechanism allows zone administrators to start using ANAME records unilaterally, without requiring secondary servers or resolvers to be upgraded.

- The resource record definition in Section 2 is intended to provide zone data portability between standards-compliant DNS servers and the common core functionality of existing proprietary ANAME-like facilities.

- The zone maintenance mechanism described in Section 5 behaves as if DNS UPDATE [RFC2136] were being used to keep an ANAME's sibling address records in sync with the ANAME target; this allows it to interoperate with existing DNSSEC signers, secondary servers, and resolvers.

This is enough to be useful by itself. However, it can be less than optimal in certain situations: for instance, when the ANAME target uses clever tricks to provide different answers to different clients to improve latency or load balancing.

- The Additional section processing rules in Section 3 inform resolvers that an ANAME record is in play.

- Resolvers can use this ANAME information as described in Section 6 to obtain answers that are tailored to the resolver rather than to the zone's primary master.

Resolver support for ANAME is not necessary, since ANAME-oblivious resolvers will get working answers from authoritative servers. It's just an optimization that can be rolled out incrementally, and that will help ANAME to work better the more widely it is deployed.

1.2. Terminology

An "address record" is a DNS resource record whose type is A or AAAA. These are referred to as "address types". "Address query" refers to a DNS query for any address type.

When talking about "address records" we mean the entire RRset, including owner name and TTL. We treat missing address records (i.e. NXDOMAIN or NODATA) the same successfully resolving as a set of zero address records, and distinct from "failure" which covers error responses such as SERVFAIL or REFUSED.

The "sibling address records" of an ANAME record are the address records at the same owner name as the ANAME, which are subject to ANAME substitution.

The "target address records" of an ANAME record are the address records obtained by resolving the ultimate target of the ANAME (see Section 4).

Other DNS-related terminology can be found in [I-D.ietf-dnsop-terminology-bis].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and

"OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. The ANAME resource record

This document defines the "ANAME" DNS resource record type, with RR TYPE value [TBD].

2.1. Presentation and wire format

The ANAME presentation format is identical to that of CNAME [RFC1033]:

```
owner ttl class ANAME target
```

The wire format is also identical to CNAME [RFC1035], except that name compression is not permitted in ANAME RDATA, per [RFC3597].

2.2. Coexistence with other types

Only one ANAME <target> can be defined per <owner>. An ANAME RRset MUST NOT contain more than one resource record.

An ANAME's sibling address records are under the control of ANAME processing (see Section 5) and are not first-class records in their own right. They MAY exist in zone files, but they can subsequently be altered by ANAME processing.

ANAME records MAY freely coexist at the same owner name with other RR types, except they MUST NOT coexist with CNAME or any other RR type that restricts the types with which it can itself coexist.

Like other types, ANAME records can coexist with DNAME records at the same owner name; in fact, the two can be used cooperatively to redirect both the owner name address records (via ANAME) and everything under it (via DNAME).

3. Additional section processing

The requirements in this section apply to both recursive and authoritative servers.

An ANAME target MAY resolve to address records via a chain of CNAME and/or ANAME records; any CNAME/ANAME chain MUST be included when adding target address records to a response's Additional section.

3.1. Address queries

When a server receives an address query for a name that has an ANAME record, the response's Additional section:

- MUST contain the ANAME record;
- MAY contain the target address records that match the query type (or the corresponding proof of nonexistence), if they are available and the target address RDATA fields differ from the sibling address RRset.

The ANAME record indicates to a client that it might wish to resolve the target address records itself. The target address records might not be available if the server is authoritative and does not include out-of-zone or non-authoritative data in its answers, or if the server is recursive and the records are not in the cache.

3.2. ANAME queries

When a server receives an query for type ANAME, there are three possibilities:

- The query resolved to an ANAME record, and the server has the target address records; any target address records SHOULD be added to the Additional section.
- The query resolved to an ANAME record, and the server does not have the target address records; any sibling address records SHOULD be added to the Additional section.
- The query did not resolve to an ANAME record; any address records with the same owner name SHOULD be added to the Additional section of the NOERROR response.

When adding address records to the Additional section, if not all address types are present and the zone is signed, the server SHOULD include a DNSSEC proof of nonexistence for the missing address types.

4. Substituting ANAME sibling address records

This process is used by both primary masters (see Section 5) and resolvers (see Section 6), though they vary in how they apply the edit described in the final step.

The following steps MUST be performed for each address type:

1. Starting at the ANAME owner, follow the chain of ANAME and/or CNAME records as far as possible to find the ultimate target.

2. If a loop is detected, continue with an empty RRset, otherwise get the ultimate target's address records. (Ignore any sibling address records of intermediate ANAMEs.)

3. Stop if resolution failed. (Note that NXDOMAIN and NODATA count as successfully resolving an empty RRset.)

4. Replace the owner of the target address records with the owner of the ANAME record. Reduce the TTL to match the ANAME record if it is greater. Drop any RRSIG records.

5. Stop if this modified RRset is the same as the sibling RRset (ignoring any RRSIG records). The comparison MAY treat nearly-equal TTLs as the same.

6. Delete the sibling address RRset and replace it with the modified RRset.

At this point, the substituted RRset is not signed. A primary master will proceed to sign the substituted RRset, whereas resolvers can only use the substituted RRset when an unsigned answer is appropriate. This is explained in more detail in the following sections.

5. ANAME processing by primary masters

Each ANAME's sibling address records are kept up-to-date as if by the following process, for each address type:

- Perform ANAME sibling address record substitution as described in Section 4. Any edit performed in the final step is applied to the ANAME's zone in the same manner as a DNS UPDATE [RFC2136].

- If resolution failed, wait for a period before trying again. This retry time SHOULD be configurable.

- Otherwise, wait until the target address record TTL has expired, then repeat.

The following informative subsections explore the effects of this specification, to clarify how it can work in practice.

5.1. Implications

A zone containing ANAME records has to be a dynamic zone, similar to automatic DNSSEC signature maintenance.

DNSSEC signatures on sibling address records are generated in the same way as for normal DNS UPDATEs.

Sibling address records are committed to the zone and stored in nonvolatile storage. This allows a server to restart without delays due to ANAME processing.

A zone containing ANAME records that point to frequently-changing targets will itself change frequently, which can increase the number of zone transfers.

Sibling address records are served from authoritative servers with a fixed TTL. Normally this TTL is expected to be the same as the target address records' TTL (or the ANAME TTL if that is smaller); however the exact mechanism for obtaining the target is unspecified, so cache effects or deliberate policies might make the sibling TTL smaller. There is a longer discussion of TTL handling in {#ttls}.

Secondary servers rely on zone transfers to obtain sibling address records, just like the rest of the zone, and serve them in the usual way (with Section 3 Additional section processing if they support it). A working DNS NOTIFY [RFC1996] setup is necessary to avoid extra delays propagating updated sibling address records when they change.

5.2. Alternatives

The process at the start of this section is specified using the mighty weasel words "as if", which are intended to allow a great deal of latitude to implementers so long as the observed behaviour is compatible.

For instance, it is likely to be more efficient to manage the polling per ANAME target rather than per ANAME as specified.

More radically, some existing ANAME-like implementations are based on a different DNS server architecture, in which a zone's published authoritative servers all perform the duties of a primary master in a distributed manner: provisioning records from a non-DNS back-end store, refreshing DNSSEC signatures, and so forth. This architecture does not use standard zone transfers, so there is no need for its ANAME implementation to poll the target address records to ensure that its secondary servers are up to date (because there are no

secondary servers as such). Instead the authoritative servers can do ANAME sibling address substitution on demand.

There are other variant architectures which use zone transfers within the provisioning system, but where the authoritative servers are able to independently vary the zone contents. They can conform to this specification provided their behaviour is consistent with it: unusual behaviour can appear "as if" there were a rapidly updating zone or multiple primary masters, etc.

The exact mechanism for obtaining the target address records is unspecified; typically they will be resolved in the DNS in the usual way, but if an ANAME implementation has special knowledge of the target it can short-cut the substitution process, or use clever tricks such as client-dependant answers.

6. ANAME processing by resolvers

When a resolver makes an address query in the usual way, it might receive a response containing ANAME information in the additional section, as described in Section 3. This informs the resolver that it MAY resolve the ANAME target address records to get answers that are tailored to the resolver rather than the ANAME's primary master. It SHOULD include the target address records in the Additional section of its responses as described in Section 3.

In order to provide tailored answers to clients that are ANAME-oblivious, the resolver MAY do its own sibling address record substitution in the following situations:

- The resolver's client queries with DO=0. (As discussed in Section 8, if the resolver finds it would downgrade a secure answer to insecure, it MAY choose not to substitute the sibling address records.)

- The resolver's client queries with DO=1 and the ANAME and sibling address records are unsigned. (Note that this situation does not apply when the records are signed but insecure: the resolver might not be able to validate them because of a broken chain of trust, but its client could have an extra trust anchor that does allow it to validate them; if the resolver substitutes the sibling address records they will become bogus.)

In these first two cases, the resolver MAY perform ANAME sibling address record substitution as described in Section 4. Any edit performed in the final step is applied to response's Answer section. The resolver SHOULD then perform Additional section processing as described in Section 3.

If the resolver's client is querying using an API such as "getaddrinfo" [RFC3493] that does not support DNSSEC validation, the resolver MAY perform ANAME sibling address record substitution as described in Section 4. Any edits performed in the final step are applied to the addresses returned by the API. (This case is for validating stub resolvers that query an upstream recursive server with DO=1, so they cannot rely on the recursive server to do ANAME substitution for them.)

7. IANA considerations

IANA is requested to assign a DNS RR TYPE value for ANAME resource records under the "Resource Record (RR) TYPEs" subregistry under the "Domain Name System (DNS) Parameters" registry.

IANA might wish to consider the creation of a registry of address types; addition of new types to such a registry would then implicitly update this specification.

8. Security considerations

When a primary master updates an ANAME's sibling address records to match its target address records, it is uses its own best information as to the correct answer. The updated records might be signed by the primary master, but that is not a guarantee of the actual correctness of the answer. This can have the effect of promoting an insecure response from the ANAME <target> to a signed response from the <owner>, which can then appear to clients to be more trustworthy than it should. To mitigate harm from this, DNSSEC validation SHOULD be used when resolving the ANAME <target>. Primary masters MAY refuse to substitute ANAME sibling address records unless the <target> node is both signed and validated.

When a resolver substitutes an ANAME's sibling address records, it can find that the sibling address records are secure but the target address records are insecure. Going ahead with the substitution will downgrade a secure answer to an insecure one. But this is likely to be the counterpart of the situation described in the previous paragraph, so the resolver is downgrading an answer that the ANAME's primary master upgraded. A resolver will only downgrade an answer in this way when its client is security-oblivious; however the client's path to the resolver is likely to be practically safer than the resolver's path to the ANAME target's servers. Resolvers MAY choose not to substitute sibling address records when they are more secure than the target address records.

9. References

9.1. Normative References

[I-D.ietf-dnsop-terminology-bis]
Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", draft-ietf-dnsop-terminology-bis-14 (work in progress), September 2018.

[RFC1033] Lottor, M., "Domain Administrators Operations Guide", RFC 1033, DOI 10.17487/RFC1033, November 1987, <https://www.rfc-editor.org/info/rfc1033>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <https://www.rfc-editor.org/info/rfc2136>.

[RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", RFC 3597, DOI 10.17487/RFC3597, September 2003, <https://www.rfc-editor.org/info/rfc3597>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[RFC0882] Mockapetris, P., "Domain names: Concepts and facilities", RFC 882, DOI 10.17487/RFC0882, November 1983, <https://www.rfc-editor.org/info/rfc882>.

[RFC0973] Mockapetris, P., "Domain system changes and observations", RFC 973, DOI 10.17487/RFC0973, January 1986, <https://www.rfc-editor.org/info/rfc973>.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <https://www.rfc-editor.org/info/rfc1996>.

[RFC2065] Eastlake 3rd, D. and C. Kaufman, "Domain Name System Security Extensions", RFC 2065, DOI 10.17487/RFC2065, January 1997, <https://www.rfc-editor.org/info/rfc2065>.

[RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS NCACHE)", RFC 2308, DOI 10.17487/RFC2308, March 1998, <https://www.rfc-editor.org/info/rfc2308>.

[RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, DOI 10.17487/RFC3493, February 2003, <https://www.rfc-editor.org/info/rfc3493>.

9.3. URIs

[1] https://github.com/each/draft-aname

Appendix A. Acknowledgments

Thanks to Mark Andrews, Ray Bellis, Stefan Buehler, Paul Ebersman, Richard Gibson, Tatuya JINMEI, Hakan Lindqvist, Mattijs Mekking, Stephen Morris, Bjorn Mott, Richard Salts, Mukund Sivaraman, Job Snijders, Jan Vcelak, Paul Vixie, Duane Wessels, and Paul Wouters for discussion and feedback.

Appendix B. Implementation status

PowerDNS currently implements a similar authoritative-only feature using "ALIAS" records, which are expanded by the primary server and transfered as address records to secondaries.

[TODO: Add discussion of DNSimple, DNS Made Easy, EasyDNS, Cloudflare, Amazon, Dyn, and Akamai.]

Appendix C. Historical note

In the early DNS [RFC0882], CNAME records were allowed to coexist with other records. However this led to coherency problems: if a resolver had no cache entries for a given name, it would resolve queries for un-cached records at that name in the usual way; once it had cached a CNAME record for a name, it would resolve queries for un-cached records using CNAME target instead.

For example, given the zone contents below, the original CNAME behaviour meant that if you asked for "alias.example.com TXT" first, you would get the answer "owner", but if you asked for "alias.example.com A" then "alias.example.com TXT" you would get the answer "target".

```
alias.example.com.      TXT    "owner"
alias.example.com.      CNAME  canonical.example.com.
canonical.example.com.  TXT    "target"
canonical.example.com.  A      192.0.2.1
```

This coherency problem was fixed in [RFC0973] which introduced the inconvenient rule that a CNAME acts as an alias for all other RR types at a name, which prevents the coexistence of CNAME with other records.

A better fix might have been to improve the cache's awareness of which records do and do not coexist with a CNAME record. However that would have required a negative cache mechanism which was not added to the DNS until later [RFC1034] [RFC2308].

While [RFC2065] relaxed the restriction by allowing coexistence of CNAME with DNSSEC records, this exception is still not applicable to other resource records. RRSIG and NSEC exist to prove the integrity of the CNAME record; they are not intended to associate arbitrary data with the domain name. DNSSEC records avoid interoperability problems by being largely invisible to security-oblivious resolvers.

Now that the DNS has negative caching, it is tempting to amend the algorithm for resolving with CNAME records to allow them to coexist with other types. Although an amended resolver will be compatible with the rest of the DNS, it will not be of much practical use because authoritative servers which rely on coexisting CNAMEs will not interoperate well with older resolvers. Practical experiments show that the problems are particularly acute when CNAME and MX try to coexist.

Appendix D. On preserving TTLs

An ANAME's sibling address records are in an unusual situation: they are authoritative data in the owner's zone, so from that point of view the owner has the last say over what their TTL should be; on the other hand, ANAMEs are supposed to act as aliases, in which case the target should control the address record TTLs.

However there are some technical constraints that make it difficult to preserve the target address record TTLs.

The conclusion of the following subsections is that the end-to-end TTL (from the authoritative servers for the target address records to end-user DNS caches) will be the target address record TTL plus the sibling address record TTL.

[MM: Discuss: I think it should be just the ANAME record TTL perhaps the minimum of ANAME and sibling address RRset TTL. We should provide some guidance on TTL settings for ANAME).

[TF: see issue #30]

D.1. Query bunching

If the times of end-user queries for a domain name are well distributed, then (normally) queries received by the authoritative servers for that domain are also well distributed. If the domain is popular, a recursive server will re-query for it once every TTL seconds, but the periodic queries from all the various recursive servers will not be aligned, so the queries remain well distributed.

However, imagine that the TTLs of an ANAME's sibling address records are decremented in the same way as cache entries in recursive servers. Then all the recursive servers querying for the name will try to refresh their caches at the same time, when the TTL reaches zero. They will become synchronized and all the queries for the domain will be bunched into periodic spikes.

This specification says that ANAME sibling address records have a normal fixed TTL derived from (e.g. equal or nearly equal to) the target address records' original TTL. There is no cache-like decrementing TTL, so there is no bunching of queries.

D.2. Upstream caches

There are two straightforward ways to get an RRset's original TTL:

- by directly querying an authoritative server;
- using the original TTL field from the RRset's RRGIG record(s).

However, not all zones are signed, and a primary master might not be able to directly query other authoritative servers (e.g. if it is a hidden primary behind a strict firewall). Instead it might have to obtain an ANAME's target address records via some other recursive server.

Querying via a separate recursive server means the primary master cannot trivially obtain the target address records' original TTLs.

Fortunately this is likely to be a self-correcting problem for similar reasons to the query-bunching discussed in the previous subsection. The primary master re-checks the target address records just after the TTL expires, when its upstream cache has just refreshed them, so the TTL will be nearly equal to the original TTL.

A related consideration is that the primary master cannot in general refresh its copies of an ANAME's target address records more frequently than their TTL, without privileged control over its resolver cache.

Combined with the requirement that sibling address records are served with a fixed TTL, this means that the end-to-end TTL will be the target address record TTL (which determines when the sibling address records are updated) plus the sibling address record TTL (which determines when end-user caches are updated).

D.3. ANAME chains

ANAME sibling address record substitution is made slightly more complicated by the requirement to follow chains of ANAME and/or CNAME records. This stops the end-to-end TTL from being inflated by each ANAME in the chain.

D.4. TTLs and zone transfers

When things are working properly (with secondary name servers responding to NOTIFY messages promptly) the authoritative servers will follow changes to ANAME target address records according to their TTLs. As a result the end-to-end TTL is unchanged from the previous subsection.

If NOTIFY doesn't work, the TTLs can be stretched by the zone's SOA refresh timer. More serious breakage can stretch them up to the zone expiry time.

Appendix E. Answer vs Additional sections

[MM: Discuss what should be in the additional section: ANAME makes sense, but differs from CNAME logic (where the CNAME is in the answer section). Additional target records that match the query type in my opinion should go in the answer section. Additional target address records that do not match the query type can go in the additional section].

[TF: from experience with DNAME I think there's a risk of interop problems if we put unexpected records in the answer section, so I

said everything should go in additional. We'll expand this appendix to explain the rationale.]

Appendix F. Changes since the last revision

[This section is to be removed before publication as an RFC.]

The full history of this draft and its issue tracker can be found at https://github.com/each/draft-aname [1]

- "-02": Major revamp, so authoritative servers (other than primary masters) now do not do any special ANAME processing, just Additional section processing.

Authors' Addresses

Tony Finch
University of Cambridge
University Information Services
Roger Needham Building
7 JJ Thomson Avenue
Cambridge CB3 0RB
England

Email: dot@dotat.at

Evan Hunt
ISC
950 Charter St
Redwood City, CA 94063
USA

Email: each@isc.org

Peter van Dijk
PowerDNS.COM B.V.
Den Haag
The Netherlands

Email: peter.van.dijk@powerdns.com

Anthony Eden
DNSimple
Boston, MA USA

Email: anthony.eden@dnsimple.com
URI: https://dnsimple.com/