Internet Engineering Task Force
Internet-Draft
Intended status: Experimental
Expires: May 11, 2019

D. Wessels
P. Barber
M. Weinberg
Verisign
W. Kumari
Google
W. Hardaker
USC/ISI

November 7, 2018

Message Digest for DNS Zones
draft-wessels-dns-zone-digest-05

Abstract

This document describes an experimental protocol and new DNS Resource Record that can be used to provide a message digest over DNS zone data. The ZONEMD Resource Record conveys the message digest data in the zone itself. When a zone publisher includes an ZONEMD record, recipients can verify the zone contents for accuracy and completeness. This provides assurance that received zone data matches published data, regardless of how the zone data has been transmitted and received.

ZONEMD is not designed to replace DNSSEC. Whereas DNSSEC protects individual RRSets (DNS data with fine granularity), ZONEMD protects protects a zone's data as a whole, whether consumed by authoritative name servers, recursive name servers, or any other applications.

As specified at this time, ZONEMD is not designed for use in large, dynamic zones due to the time and resources required for digest calculation. The ZONEMD record described in this document includes fields reserved for future work to support large, dynamic zones.

Status of This Memo

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 May 11, 2019.

Table of Contents

1. Introduction

In the DNS, a zone is the collection of authoritative resource records (RRs) sharing a common origin ([RFC7719]). Zones are often stored as files on disk in the so-called master file format [RFC1034]. Zones are generally distributed among name servers using the AXFR [RFC5936], and IXFR [RFC1995] protocols. Zone files can also be distributed outside of the DNS, with such protocols as FTP, HTTP, rsync, and even via email. Currently there is no standard way to verify the authenticity of a stand-alone zone.

This document introduces a new RR type that serves as a cryptographic message digest of the data in a zone. It allows a receiver of the zone to verify the zone's authenticity, especially when used in combination with DNSSEC. This technique makes the message digest a part of the zone itself, allowing verification the zone as a whole, no matter how it is transmitted. Furthermore, the digest is based on the wire format of zone data. Thus, it is independent of presentation format, such as changes in whitespace, capitalization, and comments.

DNSSEC provides three strong security guarantees relevant to this protocol:

1. whether or not to expect DNSSEC records in the zone,

2. whether or not to expect a ZONEMD record in a signed zone, and

3. whether or not the ZONEMD record has been altered since it was signed.

This specification is OPTIONAL to implement by both publishers and consumers of zone data.

1.1. Motivation

The motivation for this protocol enhancement is the desire for the ability to verify the authenticity of a stand-alone zone, regardless of how it is transmitted. A consumer of zone data should be able to verify that the data is as-published by the zone operator.

One approach to preventing data tampering and corruption is to secure the distribution channel. The DNS has a number of features that can already be used for channel security. Perhaps the most widely used is DNS transaction signatures (TSIG [RFC2845]). TSIG uses shared secret keys and a message digest to protect individual query and response messages. It is generally used to authenticate and validate UPDATE [RFC2136], AXFR [RFC5936], and IXFR [RFC1995] messages.

DNS Request and Transaction Signatures (SIG(0) [RFC2931]) is another protocol extension designed to authenticate individual DNS transactions. Whereas SIG records were originally designed to cover specific RR types, SIG(0) is used to sign an entire DNS message.

Unlike TSIG, SIG(0) uses public key cryptography rather than shared secrets.

The Transport Layer Security protocol suite is also designed to provide channel security. One can easily imagine the distribution of zones over HTTPS-enabled web servers, as well as DNS-over-HTTPS [dns-over-https], and perhaps even a future version of DNS-over-TLS ([RFC7858]).

Unfortunately, the protections provided by these channel security techniques are (in practice) ephemeral and are not retained after the data transfer is complete. They can ensure that the client receives the data from the expected server, and that the data sent by the server is not modified during transmission. However, they do not guarantee that the server transmits the data as originally published, and do not provide any methods to verify data that is read after transmission is complete. For example, a name server loading saved zone data upon restart cannot guarantee that the on-disk data has not

been modified. For these reasons, it is preferable to secure the data itself.

Why not simply rely on DNSSEC, which provides certain data security guarantees? Certainly for zones that are signed, a recipient could validate all of the signed RRSets. Additionally, denial-of-existence records can prove that RRSets have not been added or removed. However, not all RRSets in a zone are signed. The design of DNSSEC stipulates that delegations (non-apex NS records) are not signed, and neither are any glue records. Thus, changes to delegation and glue records cannot be detected by DNSSEC alone. Furthermore, zones that employ NSEC3 with opt-out are susceptible to the removal or addition of names between the signed nodes. Whereas DNSSEC is primarily designed to protect consumers of DNS response messages, this protocol is designed to protect consumers of zones.

There are existing tools and protocols that provide data security, such as OpenPGP [RFC4880] and S/MIME [RFC3851]. In fact, the internic.net site publishes PGP signatures along side the root zone and other files available there. However, this is a detached signature with no strong association to the corresponding zone file other than its timestamp. Non-detached signatures are, of course,

possible, but these necessarily change the format of the file being distributed. That is, a zone signed with OpenPGP or S/MIME no longer looks like a DNS zone and could not directly be loaded into a name server. Once loaded the signature data is lost, so it does not survive further propagation.

It seems the desire for data security in DNS zones was envisioned as far back as 1997. [RFC2065] is an obsoleted specification of the first generation DNSSEC Security Extensions. It describes a zone transfer signature, aka AXFR SIG, which is similar to the technique proposed by this document. That is, it proposes ordering all (signed) RRSets in a zone, hashing their contents, and then signing the zone hash. The AXFR SIG is described only for use during zone transfers. It did not postulate the need to validate zone data distributed outside of the DNS. Furthermore, its successor, [RFC2535], omits the AXFR SIG, while at the same time introducing an IXFR SIG.

1.2. Design Overview

This document introduces a new Resource Record type designed to convey a message digest of the content of a zone. The digest is calculated at the time of zone publication. Ideally the zone is signed with DNSSEC to guarantee that any modifications of the digest can be detected. The procedures for digest calculation and DNSSEC

signing are similar (i.e., both require the same ordering of RRs) and can be done in parallel.

The zone digest is designed to be used on zones that are relatively stable and have infrequent updates. As currently specified, the digest is re-calculated over the entire zone content each time. This specification does not provide an efficient mechanism for incremental updates of zone data. It does, however, reserve a field in the ZONEMD record for future work to support incremental zone digest algorithms (e.g. using Merkle trees).

It is expected that verification of a zone digest would be implemented in name server software. That is, a name server can verify the zone data it was given and refuse to serve a zone which fails verification. For signed zones, the name server needs a trust

anchor to perform DNSSEC validation. For signed non-root zones, the name server may need to send queries to validate a chain-of-trust. Digest verification could also be performed externally.

1.3. Use Cases

1.3.1. Root Zone

The root zone [InterNIC] is one of the most widely distributed DNS zone on the Internet, served by 930 separate instances [RootServers] at the time of this writing. Additionally, many organizations configure their own name servers to serve the root zone locally. Reasons for doing so include privacy and reduced access time. [RFC7706] describes one, but not the only, way to do this. As the root zone spreads beyond its traditional deployment boundaries, the need for verification of the completeness of the zone contents becomes increasingly important.

1.3.2. Providers, Secondaries, and Anycast

Since its very early days, the developers of the DNS recognized the importance of secondary name servers and service diversity. However, they may not have anticipated the complexity of modern DNS service provisioning which can include multiple third-party providers and hundreds of anycast instances. Instead of a simple primary-to-secondary zone distribution system, today it is possible to have multiple levels, multiple parties, and multiple protocols involved in the distribution of zone data. This complexity introduces new places for problems to arise. The zone digest protects the integrity of data that flows through such systems.

1.3.3. Response Policy Zones

DNS Response Policy Zones is "a method of expressing DNS response policy information inside specially constructed DNS zones..." [RPZ]. A number of companies provide RPZ feeds, which can be consumed by name server and firewall products. Since these are zones, AXFR is often, but not necessarily used for transmission. While RPZ zones can certainly be signed with DNSSEC, the data is not queried

directly, and would not be subject to DNSSEC validation.

1.3.4. Centralized Zone Data Service

ICANN operates the Centralized Zone Data Service [CZDS], which is a repository of top-level domain zone files. Users request access to the system, and to individual zones, and are then able to download zone data for certain uses. Adding a zone digest to these would provide CZDS users with assurances that the data has not been modified. Note that ZONEMD could be added to CZDS zone data independently of the zone served by production name servers.

1.3.5. General Purpose Comparison Check

Since the zone digest does not depend on presentation format, it could be used to compare multiple copies of a zone received from different sources, or copies generated by different processes.

1.4. Requirements Language

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 ZONEMD Resource Record

This section describes the ZONEMD Resource Record, including its fields, wire format, and presentation format. The Type value for the ZONEMD RR is TBD. The ZONEMD RR is class independent. The RDATA of the resource record consists of four fields: Serial, Digest Type, Reserved, and Digest.

FOR DISCUSSION: This document is currently written as though a zone MUST NOT contain more than one ZONEMD RR. Having exactly one ZONEMD record per zone simplifies this protocol and eliminates confusion around downgrade attacks, at the expense of algorithm agility.

2.1. ZONEMD RDATA Wire Format

The ZONEMD RDATA wire format is encoded as follows:

```
                        1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                             Serial                            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Digest Type  |    Reserved    |                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                              |
   |                             Digest                            |
   /                                                               /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.1.1. The Serial Field

The Serial field is a 32-bit unsigned integer in network order. It is equal to the serial number from the zone's SOA record ([RFC1035] section 3.3.13) for which the message digest was generated.

The zone's serial number is included here in order to make DNS response messages of type ZONEMD meaningful. Without the serial number, a stand-alone ZONEMD digest has no association to any particular instance of a zone.

2.1.2. The Digest Type Field

The Digest Type field is an 8-bit unsigned integer that identifies the algorithm used to construct the digest.

At the time of this writing, SHA384, with value 1, is the only Digest Type defined for ZONEMD records. The Digest Type registry is further described in Section 6.

2.1.3. The Reserved Field

The Reserved field is an 8-bit unsigned integer, which is always set to zero. This field is reserved for future work to support efficient incremental updates.

2.1.4. The Digest Field

The Digest field is a variable-length sequence of octets containing the message digest. Section 3 describes how to calculate the digest for a zone. Section 4 describes how to use the digest to verify the contents of a zone.

2.2. ZONEMD Presentation Format

The presentation format of the RDATA portion is as follows:

The Serial field MUST be represented as an unsigned decimal integer.

The Digest Type field MUST be represented as an unsigned decimal integer.

The Reserved field MUST be represented as an unsigned decimal integer set to zero.

The Digest MUST be represented as a sequence of case-insensitive hexadecimal digits. Whitespace is allowed within the hexadecimal text.

2.3. ZONEMD Example

The following example shows a ZONEMD RR.

```
example.com. 86400 IN ZONEMD 2018031500 4 0 (
    FEBE3D4CE2EC2FFA4BA99D46CD69D6D29711E55217057BEE
    7EB1A7B641A47BA7FED2DD5B97AE499FAFA4F22C6BD647DE )
```

3. Calculating the Digest

3.1. Canonical Format and Ordering

Calculation of the zone digest REQUIRES the RRs in a zone to be processed in a consistent format and ordering. Correct ordering of the zone depends on (1) ordering of owner names in the zone, (2) ordering of RRSets with the same owner name, and (3) ordering of RRs within an RRSet.

This specification adopts DNSSEC's canonical ordering for names (Section 6.1 of [RFC4034]), and canonical ordering for RRs within an RRSet (Section 6.3 of [RFC4034]). It also adopts DNSSEC's canonical RR form (Section 6.2 of [RFC4034]). However, since DNSSEC does not define a canonical ordering for RRSets having the same owner name, that ordering is defined here.

3.1.1. Order of RRSets Having the Same Owner Name

For the purposes of calculating the zone digest, RRSets having the same owner name MUST be numerically ordered, in ascending order, by their numeric RR TYPE.

3.1.2. Duplicate RRs

As stated in Section 5 of [RFC2181], it is meaningless for a zone to have multiple RRs with equal owner name, class, type, and RDATA. In the interest of consistency and interoperability, such duplicate RRs MUST NOT be included in the calculation of a zone digest.

3.2. Add ZONEMD Placeholder

In preparation for calculating the zone digest, any existing ZONEMD record at the zone apex MUST first be deleted.

FOR DISCUSSION: Should non-apex ZONEMD records be allowed in a zone? Or forbidden?

Prior to calculation of the digest, and prior to signing with DNSSEC, a placeholder ZONEMD record MUST be added to the zone apex. This serves two purposes: (1) it allows the digest to cover the Serial, Digest Type, and Reserved field values, and (2) ensures that appropriate denial-of-existence (NSEC, NSEC3) records are created if the zone is signed with DNSSEC.

It is RECOMMENDED that the TTL of the ZONEMD record match the TTL of the SOA.

In the placeholder record, the Serial field MUST be set to the current SOA Serial. The Digest Type field MUST be set to the value for the chosen digest algorithm. The Reserved field MUST be set to zero. The Digest field MUST be set to all zeroes and of length appropriate for the chosen digest algorithm.

3.3. Optionally Sign the Zone

Following addition of the placeholder record, the zone MAY be signed with DNSSEC. Note that when the digest calculation is complete, and the ZONEMD record is updated, the signature(s) for that record MUST be recalculated and updated as well. Therefore, the signer is not required to calculate a signature over the placeholder record at this step in the process, but it is harmless to do so.

3.4. Calculate the Digest

The zone digest is calculated by concatenating the canonical on-the-wire form (without name compression) of all RRs in the zone, in the order described above, subject to the inclusion/exclusion rules described below, and then applying the digest algorithm:


```
digest = digest_algorithm( RR(1) | RR(2) | RR(3) | ... )
```

where "|" denotes concatenation, and

```
RR(i) = owner | type | class | TTL | RDATA length | RDATA
```

3.4.1. Inclusion/Exclusion Rules

When calculating the digest, the following inclusion/exclusion rules apply:

- o All records in the zone, including glue records, MUST be included.
- o Occluded data ([RFC5936] Section 3.5) MUST be included.
- o Duplicate RRs with equal owner, class, type, and RDATA MUST NOT be included.
- o The placeholder ZONEMD RR MUST be included.
- o If the zone is signed, DNSSEC RRs MUST be included, except:
- o The RRSIG covering ZONEMD MUST NOT be included.

3.5. Update ZONEMD RR

Once the zone digest has been calculated, its value is then copied to the Digest field of the ZONEMD record.

If the zone is signed with DNSSEC, the appropriate RRSIG records covering the ZONEMD record MUST then be added or updated. Because the ZONEMD placeholder was added prior to signing, the zone will

already have the appropriate denial-of-existence (NSEC, NSEC3) records.

Some implementations of incremental DNSSEC signing might update the zone's serial number for each resigning. However, to preserve the calculated digest, generation of the ZONEMD signature at this time MUST NOT also result in a change of the SOA serial number.

4. Verifying Zone Message Digest

The recipient of a zone that has a message digest record can verify the zone by calculating the digest as follows:

1. The verifier SHOULD first determine whether or not to expect DNSSEC records in the zone. This can be done by examining locally configured trust anchors, or querying for (and

validating) DS RRs in the parent zone. For zones that are provably unsigned, digest validation continues at step 4 below.

2. For zones that are provably signed, the existence of the apex ZONEMD record MUST be verified. If the ZONEMD record provably does not exist, digest verification cannot be done. If the ZONEMD record does provably exist, but is not found in the zone, digest verification MUST NOT be considered successful.

3. For zones that are provably signed, the SOA RR and ZONEMD RR MUST have valid signatures, chaining up to a trust anchor. If DNSSEC validation of the SOA or ZONEMD records fails, digest verification MUST NOT be considered successful.

4. If the zone contains more than one apex ZONEMD RR, digest verification MUST NOT be considered successful.

5. The SOA Serial field MUST exactly match the ZONEMD Serial field. If the fields to not match, digest verification MUST NOT be considered successful.

6. The ZONEMD Digest Type field MUST be checked. If the verifier does not support the given digest type, it SHOULD report that the zone digest could not be verified due to an unsupported algorithm.

7. The Reserved field MUST be checked. If the Reserved field value is not zero, verification MUST NOT be considered successful.

8. The received Digest Type and Digest values are copied to a temporary location.

9. The ZONEMD RR's RDATA is reset to the placeholder values described in Section 3.2.

10. The zone digest is computed over the zone data as described in Section 3.4.

11. The calculated digest is compared to the received digest stored in the temporary location. If the two digest values match, verification is considered successful. Otherwise, verification MUST NOT be considered successful.

12. The ZONEMD RR's RDATA is reset to the received Digest Type and Digest stored in the temporary location. Thus, any downstream clients can similarly verify the zone.

5. Scope of Experimentation

This memo is published as an Experimental RFC. The purpose of the experimental period is to provide the community time to analyze and evaluate to the methods defined in this document, particularly with regard to the wide variety of DNS zones in use on the Internet.

Additionally, the ZONEMD record defined in this document includes a Reserved field in the form of an 8-bit integer. The authors have a particular future use in mind for this field, namely to support efficient digests in large, dynamic zones. We intend to conduct future experiments using Merkle trees of varying depth. The choice of tree depth can be encoded in this reserved field. We expect values for tree depth to range from 0 to 10, requiring at most four bits of this field. This leaves another four bits available for other future uses, if absolutely necessary.

FOR DISCUSSION: The authors are willing to remove the Reserved field

from this specification if the working group would prefer it. It would mean, however, that a future version of this protocol designed to efficiently support large, dynamic zones would most likely require a new RR type.

The duration of the experiment is expected to be no less than two years from the publication of this document. If the experiment is successful, it is expected that the findings of the experiment will result in an updated document for Standards Track approval.

6. IANA Considerations

6.1. ZONEMD RRtype

This document defines a new DNS RR type, ZONEMD, whose value TBD has been allocated by IANA from the "Resource Record (RR) TYPEs" subregistry of the "Domain Name System (DNS) Parameters" registry:

Type: ZONEMD

Value: TBD

Meaning: Message Digest Over Zone Data

Reference: This document

6.2. ZONEMD Digest Type

This document asks IANA to create a new "ZONEMD Digest Types" registry with initial contents as follows:

Value	Description	Status	Reference
1	SHA384	Mandatory	[RFC6605]

Table 1: ZONEMD Digest Types

7. Security Considerations

7.1. Attacks Against the Zone Digest

The zone digest allows the receiver to verify that the zone contents haven't been modified since the zone was generated/published. Verification is strongest when the zone is also signed with DNSSEC. An attacker, whose goal is to modify zone content before it is used by the victim, may consider a number of different approaches.

The attacker might perform a downgrade attack to an unsigned zone. This is why Section 4 RECOMMENDS that the verifier determine whether or not to expect DNSSEC signatures for the zone in step 1.

The attacker might perform a downgrade attack by removing the ZONEMD record. This is why Section 4 REQUIRES that the verifier checks DNSSEC denial-of-existence proofs in step 2.

The attacker might alter the Digest Type or Digest fields of the ZONEMD record. Such modifications are detectable only with DNSSEC validation.

7.2. Attacks Utilizing the Zone Digest

Nothing in this specification prevents clients from making, and servers from responding to, ZONEMD queries. One might consider how well ZONEMD responses could be used in a distributed denial-of-service amplification attack.

The ZONEMD RR is moderately sized, much like the DS RR. A single ZONEMD RR contributes approximately 40 to 65 octets to a DNS response, for currently defined digest types. Certainly other query types result in larger amplification effects (i.e., DNSKEY).

8. Privacy Considerations

This specification has no impacts on user privacy.

9. Acknowledgments

The authors wish to thank David Blacka, Scott Hollenbeck, and Rick Wilhelm for providing feedback on early drafts of this document. Additionally, they thank Joe Abley, Mark Andrews, Olafur Gudmundsson, Paul Hoffman, Evan Hunt, Shumon Huque, Tatuya Jinmei, Burt Kaliski, Shane Kerr, Matt Larson, John Levine, Ed Lewis, Mukund Sivaraman, Petr Spacek, Ondrej Sury, Florian Weimer, Tim Wicinksi, Paul Wouters, and other members of the dnsop working group for their input.

10. Implementation Status

10.1. Authors' Implementation

The authors have an open source implementation in C, using the ldns library [ldns-zone-digest]. This implementation is able to perform the following functions:

- o Read an input zone and output a zone with the ZONEMD placeholder.
- o Compute zone digest over signed zone and update the ZONEMD record.
- o Re-compute DNSSEC signature over the ZONEMD record.
- o Verify the zone digest from an input zone.

This implementation does not:

- o Perform DNSSEC validation of the ZONEMD record.
- o Support the Gost digest algorithm.
- o Output the ZONEMD record in its defined presentation format.

10.2. Shane Kerr's Implementation

Shane Kerr wrote an implementation of this specification during the IETF 102 hackathon [ZoneDigestHackathon]. This implementation is in Python and is able to perform the following functions:

- o Read an input zone and a output zone with ZONEMD record.
- o Verify the zone digest from an input zone.

- Output the ZONEMD record in its defined presentation format.

- Generate Gost digests.

This implementation does not:

- Re-compute DNSSEC signature over the ZONEMD record.

- Perform DNSSEC validation of the ZONEMD record.

11. Change Log

RFC Editor: Please remove this section.

This section lists substantial changes to the document as it is being worked on.

From -00 to -01:

- Removed requirement to sort by RR CLASS.

- Added Kumari and Hardaker as coauthors.

- Added Change Log section.

- Minor clarifications and grammatical edits.

From -01 to -02:

- Emphasize desire for data security over channel security.

- Expanded motivation into its own subsection.

- Removed discussion topic whether or not to include serial in ZONEMD.

- Clarified that a zone's NS records always sort before the SOA record.

- Clarified that all records in the zone must are digested, except as specified in the exclusion rules.

- Added for discussion out-of-zone and occluded records.

- Clarified that update of ZONEMD signature must not cause a serial number change.

o Added persons to acknowledgments.

From -02 to -03:

o Added recommendation to set ZONEMD TTL to SOA TTL.

o Clarified that digest input uses uncompressed names.

o Updated Implementations section.

o Changed intended status from Standards Track to Experimental and added Scope of Experiment section.

o Updated Motivation, Introduction, and Design Overview sections in response to working group discussion.

o Gave ZONEMD digest types their own status, separate from DS digest types. Request IANA to create a registry.

o Added Reserved field for future work supporting dynamic updates.

o Be more rigorous about having just ONE ZONEMD record in the zone.

o Expanded use cases.

From -03 to -04:

o Added an appendix with example zones and digests.

o Clarified that only apex ZONEMD RRs shall be processed.

From -04 to -05:

o Made SHA384 the only supported ZONEMD digest type.

o Disassociated ZONEMD digest types from DS digest types.

o Updates to Introduction based on list feedback.

o Changed "zone file" to "zone" everywhere.

o Restored text about why ZONEMD has a Serial field.

o Clarified ordering of RRSets having same owner to be numerically ascending.

o Clarified that all duplicate RRs (not just SOA) must be suppressed in digest calculation.

o Clarified that the Reserved field must be set to zero and checked for zero in verification.

o Clarified that occluded data must be included.

o Clarified procedure for verification, using temporary location for received digest.

o Explained why Reserved field is 8-bits.

o IANA Considerations section now more specific.

o Added complex zone to examples.

o

12. References

12.1. Normative References

[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>.

[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>.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, DOI 10.17487/RFC2181, July 1997, <https://www.rfc-editor.org/info/rfc2181>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/info/rfc4034>.

[RFC6605] Hoffman, P. and W. Wijngaards, "Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC", RFC 6605, DOI 10.17487/RFC6605, April 2012, <https://www.rfc-editor.org/info/rfc6605>.

[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>.

12.2. Informative References

[CZDS] Internet Corporation for Assigned Names and Numbers, "Centralized Zone Data Service", October 2018, <https://czds.icann.org/>.

[dns-over-https] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", draft-ietf-doh-dns-over-https-12 (work in progress), June 2018, <https://tools.ietf.org/html/draft-ietf-doh-dns-over-https-12>.

[InterNIC] ICANN, "InterNIC FTP site", May 2018, <ftp://ftp.internic.net/domain/>.

[ldns-zone-digest] Verisign, "Implementation of Message Digests for DNS Zones using the ldns library", July 2018, <https://github.com/verisign/ldns-zone-digest>.

[RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995,

DOI 10.17487/RFC1995, August 1996,
<https://www.rfc-editor.org/info/rfc1995>.

[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>.

[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>.

[RFC2535] Eastlake 3rd, D., "Domain Name System Security Extensions", RFC 2535, DOI 10.17487/RFC2535, March 1999, <https://www.rfc-editor.org/info/rfc2535>.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <https://www.rfc-editor.org/info/rfc2845>.

[RFC2931] Eastlake 3rd, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, DOI 10.17487/RFC2931, September 2000, <https://www.rfc-editor.org/info/rfc2931>.

[RFC3851] Ramsdell, B., Ed., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, DOI 10.17487/RFC3851, July 2004, <https://www.rfc-editor.org/info/rfc3851>.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, DOI 10.17487/RFC4880, November 2007, <https://www.rfc-editor.org/info/rfc4880>.

[RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <https://www.rfc-editor.org/info/rfc5936>.

[RFC7706] Kumari, W. and P. Hoffman, "Decreasing Access Time to Root

Servers by Running One on Loopback", RFC 7706, DOI 10.17487/RFC7706, November 2015, <https://www.rfc-editor.org/info/rfc7706>.

[RFC7719] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", RFC 7719, DOI 10.17487/RFC7719, December 2015, <https://www.rfc-editor.org/info/rfc7719>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RootServers] Root Server Operators, "Root Server Technical Operations", July 2018, <https://www.root-servers.org/>.

[RPZ] Vixie, P. and V. Schryver, "DNS Response Policy Zones (RPZ)", draft-vixie-dnsop-dns-rpz-00 (work in progress), June 2018, <https://tools.ietf.org/html/draft-vixie-dnsop-dns-rpz-00>.

[ZoneDigestHackathon] Kerr, S., "Prototype implementation of ZONEMD for the IETF 102 hackathon in Python", July 2018, <https://github.com/shane-kerr/ZoneDigestHackathon>.

Appendix A. Example Zones With Digests

This appendex contains example zones with accurate ZONEMD records. These can be used to verify an implementation of the zone digest protocol.

A.1. Simple EXAMPLE Zone

Here, the EXAMPLE zone contains an SOA record, NS and glue records, and a ZONEMD record.

```
example.      86400   IN   SOA     ns1 admin 2018031900 (
```

```
                                     1800 900 604800 86400 )
                 86400   IN  NS       ns1
                 86400   IN  NS       ns2
                 86400   IN  ZONEMD   2018031900 1 0 (
                                     f32765ce15c50477
                                     42a08be15d9a0efb
                                     749417eaadcfa28b
                                     1bf751b6bc49f9be
                                     a615c4a386cfd6a5
                                     d85e2d2182691249 )
   ns1           3600    IN  A        127.0.0.1
   ns2           3600    IN  AAAA     ::1
```

A.2. Complex EXAMPLE Zone

Here, the EXAMPLE zone contains duplicate RRs, and an occluded RR, and one out-of-zone RR.


```
   example.      86400   IN  SOA      ns1 admin 2018031900 (
                                     1800 900 604800 86400 )
                 86400   IN  NS       ns1
                 86400   IN  NS       ns2
                 86400   IN  ZONEMD   2018031900 1 0 (
```

```
                                   686a6d74d5638612
                                   64ea4e6cc12c22d1
                                   7ebc529791d393bd
                                   e164a45390f714e9
                                   9ede0d05a5644573
                                   da4bbcc83744acf2 )
ns1             3600    IN   A     127.0.0.1
ns2             3600    IN   AAAA  ::1
occluded.sub    7200    IN   TXT   "I'm occluded but must be digested"
sub             7200    IN   NS    ns1
duplicate       300     IN   TXT   "I must be digested just once"
duplicate       300     IN   TXT   "I must be digested just once"
foo.test.       555     IN   TXT   "out-of-zone data must be excluded"
```

A.3. The URI.ARPA Zone

The URI.ARPA zone retrieved 2018-10-21.

```
; <<>> DiG 9.9.4 <<>> @lax.xfr.dns.icann.org uri.arpa axfr
; (2 servers found)
;; global options: +cmd
uri.arpa.           3600    IN      SOA     sns.dns.icann.org. (
    noc.dns.icann.org. 2018100702 10800 3600 1209600 3600 )
uri.arpa.           3600    IN      RRSIG   NSEC 8 2 3600 (
    20181028142623 20181007205525 47155 uri.arpa.
    eEC4w/oXLR1Epwgv4MBiDtSBsXhqrJVvJWUpbX8XpetAvD35bxwNCUTi
    /pAJVUXefegWeiriD2rkTgCBCMmn7YQIm3gdR+HjY/+o3BXNQnz97f+e
    HAE9EDDzoNVfL1PyV/2fde9tDeUuAGVVwmD399NGq9jWYMRpyri2kysr q/g= )
uri.arpa.           86400   IN      RRSIG   NS 8 2 86400 (
    20181028172020 20181007175821 47155 uri.arpa.
    ATyV2A2A8ZoggC+68u4GuP5MOUuR+2rr3eWOkEU55zAHld/7FiBxl4ln
    4byJYy7NudUwlMOEXajqFZE7DVl8PpcvrP3HeeGaVzKqaWj+aus0jbKF
    Bsvs2b1qDZemBfkz/IfAhUTJKnto0vSUicJKfItu0GjyYNJCz2CqEuGD Wxc= )
uri.arpa.           600     IN      RRSIG   MX 8 2 600 (
    20181028170556 20181007175821 47155 uri.arpa.
    e7/r3KXDohX1lyVavetFFObp8fB8aXT76HnN9KCQDxSnSghNM83UQV0t
    lTtD8JVeN1mCvcNFZpagwIgB7XhTtm6Beur/m5ES+4uSnVeS6Q66HBZK
    A3mR95IpevuVIZvvJ+GcCAQpBo6KRODYvJ/c/ZG6sfYWkZ7qg/Em5/+3 4UI= )
uri.arpa.           3600    IN      RRSIG   DNSKEY 8 2 3600 (
    20181028152832 20181007175821 15796 uri.arpa.
    nzpbnh0OqsgBBP8St28pLvPEQ3wZAUdEBuUwil+rtjjWlYYiqjPxZ286
    XF4Rq1usfV5x71jZz5IqswOaQgia91ylodFpLuXD6FTGs2nXGhNKkg1V
```

```
        chHgtwj70mXU72GefVgo8TxrFYzxuEFP5ZTP92t97FVWVVyyFd86sbbR
        6DZj3uA2wEvqBVLECgJLrMQ9Yy7MueJl3UA4h4E6zO2JY9Yp0W9woq0B
        dqkkwYTwzogyYffPmGAJG91RJ2h6cHtFjEZe2MnaY2glqniZ0WT9vXXd
        uFPm0KD9U77Ac+ZtctAF9tsZwSdAoL365E2L1usZbA+K0BnPPqGFJRJk
        5R0A1w== )
uri.arpa.          3600    IN      RRSIG   DNSKEY 8 2 3600 (
        20181028152832 20181007175821 55480 uri.arpa.
        lWtQV/5szQjkXmbcD47/+rOW8kJPksRFHlzxxmzt906+DBYyfrH6uq5X
        nHvrUlQO6M12uhqDeL+bDFVgqSpNy+42/OaZvaK3J8EzPZVBHPJykKMV
        63T83aAiJrAyHzOaEdmzLCpalqcEE2ImzlLHSafManRfJL8Yuv+JDZFj
        2WDWfEcUuwkmIZWX11zxp+DxwzyUlRl7x4+ok5iKZWIg5UnBAf6B8T75
        WnXzlhCw3F2pXI0a5LYg71L3Tp/xhjN6Yy9jGlIRf5BjB59X2zra3a2R
        PkI09SSnuEwHyF1mDaV5BmQrLGRnCjvwXA7ho2m+vv4SP5dUdXf+GTeA
        1HeBfw== )
uri.arpa.          3600    IN      RRSIG   SOA 8 2 3600 (
        20181029114753 20181008222815 47155 uri.arpa.
        qn8yBNoHDjGdT79U2Wu9IIahoS0YPOgYP8lG+qwPcrZ1BwGiHywuoUa2
        Mx6BWZlg+HDyaxj2iOmox+IIqoUHhXUbO7IUkJFlgrOKCgAR2twDHrXu
        9BUQHy9SoV16wYm3kBTEPyxW5FFm8vcdnKAF7sxSY8BbaYNpRIEjDx4A JUc= )
uri.arpa.          3600    IN      NSEC    ftp.uri.arpa. NS SOA (
        MX RRSIG NSEC DNSKEY )
uri.arpa.          86400   IN      NS      a.iana-servers.net.
uri.arpa.          86400   IN      NS      b.iana-servers.net.
uri.arpa.          86400   IN      NS      c.iana-servers.net.
uri.arpa.          86400   IN      NS      ns2.lacnic.net.
uri.arpa.          86400   IN      NS      sec3.apnic.net.
uri.arpa.          600     IN      MX      10 pechora.icann.org.
uri.arpa.          3600    IN      DNSKEY  256 3 8 (
        AwEAAcBi7tSart2J599zbYWspMNGN70IBWb4ziqyQYH9MTB/VCz6WyUK
        uXunwiJJbbQ3bcLqTLWEw134B6cTMHrZpjTAb5WAwg4XcWUu8mdcPTiL
        Bl6qVRlRD0WiFCTzuYUfkwsh1Rbr7rvrxSQhF5rh71zSpwV5jjjp65Wx
        SdJjlH0B )
uri.arpa.          3600    IN      DNSKEY  257 3 8 (
        AwEAAbNVv6ulgRdO31MtAehz7j3ALRjwZglWesnzvllQl/+hBRZr9QoY
        cO2I+DkO4Q1NKxox4DUIxj8SxPO3GwDuOFR9q2/CFi2O0mZjafbdYtWc
        3zSdBbi3q0cwCIx7GuG9eqlL+pg7mdk9dgdNZfHwB0LnqTD8ebLPsrO/
        Id7kBaiqYOfMlZnh2fp+2h6OOJZHtY0DK1UlssyB5PKsE0tVzo5s6zo9
        iXKe5u+8WTMaGDY49vG80JPAKE7ezMiH/NZcUMiE0PRZ8D3foq2dYuS5
        ym+vA83Z7v8A+Rwh4UGnjxKB8zmr803V0ASAmHz/gwH5Vb0nH+LObwFt
        l3wpbp+Wpm8= )
uri.arpa.          3600    IN      DNSKEY  257 3 8 (
        AwEAAbwnFTakCvaUKsXji4mgmxZUJi1IygbnGahbkmFEa0L16J+TchKR
        wcgzVfsxUGa2MmeA4hgkAooC3uy+tTmoMsgy8uq/JAj24DjiHzd46LfD
        FK/qMidVqFpYSHeq2Vv5ojkuIsx4oe4KsafGWYNOczKZgH5loGjN2aJG
        mrIm++XCphOskgCsQYl65MIzuXffzJyxlAuts+ecAIiVeqRaqQfr8LRU
        7wIsLxinXirprtQrbor+EtvlHp9qXE6ARTZDzf4jvsNpKvLFZtmxzFf3
        e/UJz5eHjpwDSiZL7xE8aE1o1nGfPtJx9ZnB3bapltaJ5wY+5XOCKgY0
        xmJVvNQlwdE= )
```

```
   ftp.uri.arpa.     3600    IN      RRSIG   NSEC 8 3 3600 (
       20181028080856 20181007175821 47155 uri.arpa.
       HClGAqPxzkYkAT7Q/QNtQeB6YrkP6EPOef+9Qo5/2zngwAewXEAQiyF9
       jD1USJiroM11QqBS3v3aIdW/LXORs4Ez3hLcKNO1cKHsOuWAqzmE+BPP
       Arfh8N95jqh/q6vpaB9UtMkQ53tM2fYU1GszOLN0knxbHgDHAh2axMGH lqM= )
   ftp.uri.arpa.     604800  IN      RRSIG   NAPTR 8 3 604800 (
       20181028103644 20181007205525 47155 uri.arpa.
       WoLi+vZzkxaoLr2IGZnwkRvcDf6KxiWQd1WZP/U+AWnV+7MiqsWPZaf0
       9toRErerGoFOiOASNxZjBGJrRgjmavOM9U+LZSconP9zrNFd4dIu6kp5
       YxlQJ0uHOvx1ZHFCj6lAt1ACUIw04ZhMydTmi27c8MzEOMepvn7iH7r7 k7k= )
   ftp.uri.arpa.     3600    IN      NSEC    http.uri.arpa. NAPTR (
       RRSIG NSEC )
   ftp.uri.arpa.     604800  IN      NAPTR   0 0 "" "" (
       "!^ftp://([^:/?#]*).*$!\\1!i" . )
   http.uri.arpa.    3600    IN      RRSIG   NSEC 8 3 3600 (
       20181029010647 20181007175821 47155 uri.arpa.
       U03NntQ73LHWpfLmUK8nMsqkwVsOGW2KdsyuHYAjqQSZvKbtmbv7HBmE
       H1+Ii3Z+wtfdMZBy5aC/6sHdx69BfZJs16xumycMlAy6325DKTQbIMN+
       ift9GrKBC7cgCd2msF/uzSrYxxg4MJQzBPvlkwXnY3b7eJSlIXisBIn7 3b8= )
   http.uri.arpa.    604800  IN      RRSIG   NAPTR 8 3 604800 (
       20181029011815 20181007205525 47155 uri.arpa.
       T7mRrdag+WSmG+n22mtBSQ/0Y3v+rdDnfQV90LN5Fq32N5K2iYFajF7F
       Tp56oOznytfcL4fHrqOE0wRc9NWOCCUec9C7Wa1gJQcllEvgoAM+L6f0
       RsEjWq6+9jvlLKMXQv0xQuMX17338uoD/xiAFQSnDbiQKxwWMqVAimv5 7Zs= )
   http.uri.arpa.    3600    IN      NSEC    mailto.uri.arpa. NAPTR (
       RRSIG NSEC )
   http.uri.arpa.    604800  IN      NAPTR   0 0 "" "" (
       "!^http://([^:/?#]*).*$!\\1!i" . )
   mailto.uri.arpa.  3600    IN      RRSIG   NSEC 8 3 3600 (
       20181028110727 20181007175821 47155 uri.arpa.
       GvxzVL85rEukwGqtuLxek9ipwjBMfTOFIEyJ7afC8HxVMs6mfFa/nEM/
       IdFvvFg+lcYoJSQYuSAVYFl3xPbgrxVSLK125QutCFMdC/YjuZEnq5cl
       fQciMRD7R3+znZfm8d8u/snLV9w4D+lTBZrJJUBe1Efc8vum5vvV7819 ZoY= )
   mailto.uri.arpa.  604800  IN      RRSIG   NAPTR 8 3 604800 (
       20181028141825 20181007205525 47155 uri.arpa.
       MaADUgc3fc5v++M0YmqjGk3jBdfIA5RuP62hUSlPsFZO4k37erjIGCfF
       j+g84yc+QgbSde0PQHszl9fE/+SU5ZXiS9YdcbzSZxp2erFpZOTchrpg
       916T4vx6i59scodjb0l6bDyZ+mtIPrc1w6b4hUyOUTsDQoAJYxdfEuMg Vy4= )
   mailto.uri.arpa.  3600    IN      NSEC    urn.uri.arpa. NAPTR (
       RRSIG NSEC )
   mailto.uri.arpa.  604800  IN      NAPTR   0 0 "" "" (
       "!^mailto:(.*)@(.*)$!\\2!i" . )
```

```
   urn.uri.arpa.     3600    IN      RRSIG   NSEC 8 3 3600 (
       20181028123243 20181007175821 47155 uri.arpa.
       Hgsw4Deops1O8uWyELGe6hpR/OEqCnTHvahlwiQkHhO5CSEQrbhmFAWe
       UOkmGAdTEYrSz+skLRQuITRMwzyFf4oUkZihGyhZyzHbcxWfuDc/Pd/9
       DSl56gdeBwy1evn5wBTms8yWQVkNtphbJH395gRqZuaJs3LD/qTyJ5Dp LvA= )
   urn.uri.arpa.     604800  IN      RRSIG   NAPTR 8 3 604800 (
```



```
       20181029071816 20181007205525 47155 uri.arpa.
       ALIZD0vBqAQQt40GQ0Efaj8OCyE9xSRJRdyvyn/H/wZVXFRFKrQYrLAS
       D/K7q6CMTOxTRCu2J8yes63WJiaJEdnh+dscXzZkmOg4n5PsgZbkvUSW
       BiGtxvz5jNncM0xVbkjbtByrvJQAO1cU1mnlDKe1FmVB1uLpVdA9Ib4J hMU= )
   urn.uri.arpa.     3600    IN      NSEC    uri.arpa. NAPTR RRSIG (
       NSEC )
   urn.uri.arpa.     604800  IN      NAPTR   0 0 "" "" (
       "/urn:([^:]+)/\\1/i" . )
   uri.arpa.         3600    IN      SOA     sns.dns.icann.org. (
       noc.dns.icann.org. 2018100702 10800 3600 1209600 3600 )
   ;; Query time: 66 msec
   ;; SERVER: 192.0.32.132#53(192.0.32.132)
   ;; WHEN: Sun Oct 21 20:39:28 UTC 2018
   ;; XFR size: 34 records (messages 1, bytes 3941)
   uri.arpa.       3600    IN      ZONEMD  2018100702 1 0 (
       80af7afd9540ff2c4c559f0d2b83393386304e093e0e66787378b2
       a578297b49b4dccb422bce2c300bb92b354575283a )
```

A.4. The ROOT-SERVERS.NET Zone

The ROOT-SERVERS.NET zone retrieved 2018-10-21.

```
   root-servers.net.     3600000 IN  SOA     a.root-servers.net. (
       nstld.verisign-grs.com. 2018091100 14400 7200 1209600 3600000 )
   root-servers.net.     3600000 IN  NS      a.root-servers.net.
   root-servers.net.     3600000 IN  NS      b.root-servers.net.
   root-servers.net.     3600000 IN  NS      c.root-servers.net.
   root-servers.net.     3600000 IN  NS      d.root-servers.net.
   root-servers.net.     3600000 IN  NS      e.root-servers.net.
   root-servers.net.     3600000 IN  NS      f.root-servers.net.
   root-servers.net.     3600000 IN  NS      g.root-servers.net.
   root-servers.net.     3600000 IN  NS      h.root-servers.net.
   root-servers.net.     3600000 IN  NS      i.root-servers.net.
   root-servers.net.     3600000 IN  NS      j.root-servers.net.
   root-servers.net.     3600000 IN  NS      k.root-servers.net.
   root-servers.net.     3600000 IN  NS      l.root-servers.net.
   root-servers.net.     3600000 IN  NS      m.root-servers.net.
   a.root-servers.net.   3600000 IN  AAAA    2001:503:ba3e::2:30
   a.root-servers.net.   3600000 IN  A       198.41.0.4
   b.root-servers.net.   3600000 IN  MX      20 mail.isi.edu.
   b.root-servers.net.   3600000 IN  AAAA    2001:500:200::b
   b.root-servers.net.   3600000 IN  A       199.9.14.201
   c.root-servers.net.   3600000 IN  AAAA    2001:500:2::c
   c.root-servers.net.   3600000 IN  A       192.33.4.12
   d.root-servers.net.   3600000 IN  AAAA    2001:500:2d::d
   d.root-servers.net.   3600000 IN  A       199.7.91.13
   e.root-servers.net.   3600000 IN  AAAA    2001:500:a8::e
   e.root-servers.net.   3600000 IN  A       192.203.230.10
   f.root-servers.net.   3600000 IN  AAAA    2001:500:2f::f
   f.root-servers.net.   3600000 IN  A       192.5.5.241
```

```
g.root-servers.net.   3600000 IN  AAAA    2001:500:12::d0d
g.root-servers.net.   3600000 IN  A       192.112.36.4
h.root-servers.net.   3600000 IN  AAAA    2001:500:1::53
h.root-servers.net.   3600000 IN  A       198.97.190.53
i.root-servers.net.   3600000 IN  MX      10 mx.i.root-servers.org.
i.root-servers.net.   3600000 IN  AAAA    2001:7fe::53
i.root-servers.net.   3600000 IN  A       192.36.148.17
j.root-servers.net.   3600000 IN  AAAA    2001:503:c27::2:30
j.root-servers.net.   3600000 IN  A       192.58.128.30
k.root-servers.net.   3600000 IN  AAAA    2001:7fd::1
k.root-servers.net.   3600000 IN  A       193.0.14.129
l.root-servers.net.   3600000 IN  AAAA    2001:500:9f::42
l.root-servers.net.   3600000 IN  A       199.7.83.42
m.root-servers.net.   3600000 IN  AAAA    2001:dc3::35
m.root-servers.net.   3600000 IN  A       202.12.27.33
root-servers.net.     3600000 IN  SOA     a.root-servers.net. (
    nstld.verisign-grs.com. 2018091100 14400 7200 1209600 3600000 )
root-servers.net.     3600000 IN  ZONEMD  2018091100 1 0 (
    aadf7a017bccd8cefe6040494800249fd5edc3d49e2e8ce8db7522f47f
    97f168db794bf5f679fbe0c8433fb66f7a0c26 )
```


Authors' Addresses

Duane Wessels
Verisign
12061 Bluemont Way
Reston, VA 20190

Phone: +1 703 948-3200
Email: dwessels@verisign.com
URI: http://verisign.com

Piet Barber
Verisign
12061 Bluemont Way
Reston, VA 20190

Phone: +1 703 948-3200
Email: pbarber@verisign.com
URI: http://verisign.com

Matt Weinberg
Verisign
12061 Bluemont Way
Reston, VA 20190

Phone: +1 703 948-3200
Email: mweinberg@verisign.com
URI: http://verisign.com

Warren Kumari
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043

Email: warren@kumari.net

Wes Hardaker
USC/ISI
P.O. Box 382
Davis, CA 95617

Email: ietf@hardakers.net