            Minimally Covering NSEC Records and DNSSEC On-line Signing

Status of This Memo

Copyright Notice

Abstract

   This document describes how to construct DNSSEC NSEC resource records
   that cover a smaller range of names than called for by RFC 4034. By
   generating and signing these records on demand, authoritative name
   servers can effectively stop the disclosure of zone contents
   otherwise made possible by walking the chain of NSEC records in a
   signed zone.

Table of Contents

1. Introduction

   With DNSSEC [1], an NSEC record lists the next instantiated name in
   its zone, proving that no names exist in the "span" between the
   NSEC's owner name and the name in the "next name" field. In this
   document, an NSEC record is said to "cover" the names between its
   owner name and next name.

Through repeated queries that return NSEC records, it is possible to
retrieve all of the names in the zone, a process commonly called
"walking" the zone. Some zone owners have policies forbidding zone
transfers by arbitrary clients; this side effect of the NSEC
architecture subverts those policies.

This document presents a way to prevent zone walking by constructing
NSEC records that cover fewer names. These records can make zone
walking take approximately as many queries as simply asking for all
possible names in a zone, making zone walking impractical. Some of
these records must be created and signed on demand, which requires
on-line private keys. Anyone contemplating use of this technique is
strongly encouraged to review the discussion of the risks of on-line
signing in Section 5.

1.2. Keywords

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in RFC 2119 [4].

2. Applicability of This Technique

The technique presented here may be useful to a zone owner that wants
to use DNSSEC, is concerned about exposure of its zone contents via
zone walking, and is willing to bear the costs of on-line signing.

As discussed in Section 5, on-line signing has several security
risks, including an increased likelihood of private keys being
disclosed and an increased risk of denial of service attack. Anyone
contemplating use of this technique is strongly encouraged to review
the discussion of the risks of on-line signing in Section 5.

Furthermore, at the time this document was published, the DNSEXT
working group was actively working on a mechanism to prevent zone
walking that does not require on-line signing (tentatively called
NSEC3). The new mechanism is likely to expose slightly more
information about the zone than this technique (e.g., the number of
instantiated names), but it may be preferable to this technique.

3. Minimally Covering NSEC Records

This mechanism involves changes to NSEC records for instantiated
names, which can still be generated and signed in advance, as well as
the on-demand generation and signing of new NSEC records whenever a
name must be proven not to exist.

In the "next name" field of instantiated names' NSEC records, rather
than list the next instantiated name in the zone, list any name that
falls lexically after the NSEC's owner name and before the next
instantiated name in the zone, according to the ordering function in
RFC 4034 [2] Section 6.1. This relaxes the requirement in Section
4.1.1 of RFC 4034 that the "next name" field contains the next owner
name in the zone. This change is expected to be fully compatible
with all existing DNSSEC validators. These NSEC records are returned
whenever proving something specifically about the owner name (e.g.,
that no resource records of a given type appear at that name).

Whenever an NSEC record is needed to prove the non-existence of a
name, a new NSEC record is dynamically produced and signed. The new
NSEC record has an owner name lexically before the QNAME but
lexically following any existing name and a "next name" lexically
following the QNAME but before any existing name.

The generated NSEC record's type bitmap MUST have the RRSIG and NSEC
bits set and SHOULD NOT have any other bits set. This relaxes the
requirement in Section 2.3 of RFC4035 that NSEC RRs not appear at
names that did not exist before the zone was signed.

The functions to generate the lexically following and proceeding
names need not be perfect or consistent, but the generated NSEC
records must not cover any existing names. Furthermore, this
technique works best when the generated NSEC records cover as few
names as possible. In this document, the functions that generate the
nearby names are called "epsilon" functions, a reference to the
mathematical convention of using the greek letter epsilon to
represent small deviations.

An NSEC record denying the existence of a wildcard may be generated
in the same way. Since the NSEC record covering a non-existent
wildcard is likely to be used in response to many queries,
authoritative name servers using the techniques described here may
want to pregenerate or cache that record and its corresponding RRSIG.

For example, a query for an A record at the non-instantiated name
example.com might produce the following two NSEC records, the first
denying the existence of the name example.com and the second denying
the existence of a wildcard:

        exampld.com 3600 IN NSEC example-.com (RRSIG NSEC)

        \).com 3600 IN NSEC +.com (RRSIG NSEC)

Before answering a query with these records, an authoritative server
must test for the existence of names between these endpoints. If the
generated NSEC would cover existing names (e.g., exampldd.com or
*bizarre.example.com), a better epsilon function may be used or the
covered name closest to the QNAME could be used as the NSEC owner
name or next name, as appropriate. If an existing name is used as
the NSEC owner name, that name's real NSEC record MUST be returned.
Using the same example, assuming an exampldd.com delegation exists,
this record might be returned from the parent:

        exampldd.com 3600 IN NSEC example-.com (NS DS RRSIG NSEC)

Like every authoritative record in the zone, each generated NSEC
record MUST have corresponding RRSIGs generated using each algorithm
(but not necessarily each DNSKEY) in the zone's DNSKEY RRset, as
described in RFC 4035 [3] Section 2.2. To minimize the number of
signatures that must be generated, a zone may wish to limit the
number of algorithms in its DNSKEY RRset.

4. Better Epsilon Functions

Section 6.1 of RFC 4034 defines a strict ordering of DNS names.
Working backward from that definition, it should be possible to
define epsilon functions that generate the immediately following and
preceding names, respectively. This document does not define such
functions. Instead, this section presents functions that come
reasonably close to the perfect ones. As described above, an
authoritative server should still ensure than no generated NSEC
covers any existing name.

To increment a name, add a leading label with a single null (zero-
value) octet.

To decrement a name, decrement the last character of the leftmost
label, then fill that label to a length of 63 octets with octets of
value 255. To decrement a null (zero-value) octet, remove the octet
-- if an empty label is left, remove the label. Defining this
function numerically: fill the leftmost label to its maximum length
with zeros (numeric, not ASCII zeros) and subtract one.

In response to a query for the non-existent name foo.example.com,
these functions produce NSEC records of the following:

```
   fon\255\255\255\255\255\255\255\255\255\255\255\255\255\255
   \255\255\255\255\255\255\255\255\255\255\255\255\255\255\255
   \255\255\255\255\255\255\255\255\255\255\255\255\255\255\255
   \255\255\255\255\255\255\255\255\255\255\255\255\255\255\255
   \255.example.com 3600 IN NSEC \000.foo.example.com ( NSEC RRSIG )

   \)\255\255\255\255\255\255\255\255\255\255\255\255\255\255
   \255\255\255\255\255\255\255\255\255\255\255\255\255\255\255
   \255\255\255\255\255\255\255\255\255\255\255\255\255\255\255
   \255\255\255\255\255\255\255\255\255\255\255\255\255\255\255
   \255\255.example.com 3600 IN NSEC \000.*.example.com ( NSEC RRSIG )
```

   The first of these NSEC RRs proves that no exact match for
   foo.example.com exists, and the second proves that there is no
   wildcard in example.com.

   Both of these functions are imperfect: they do not take into account
   constraints on number of labels in a name nor total length of a name.
   As noted in the previous section, though, this technique does not
   depend on the use of perfect epsilon functions: it is sufficient to
   test whether any instantiated names fall into the span covered by the
   generated NSEC and, if so, substitute those instantiated owner names
   for the NSEC owner name or next name, as appropriate.

5. Security Considerations

   This approach requires on-demand generation of RRSIG records. This
   creates several new vulnerabilities.

   First, on-demand signing requires that a zone's authoritative servers
   have access to its private keys. Storing private keys on well-known
   Internet-accessible servers may make them more vulnerable to
   unintended disclosure.

   Second, since generation of digital signatures tends to be
   computationally demanding, the requirement for on-demand signing
   makes authoritative servers vulnerable to a denial of service attack.

   Last, if the epsilon functions are predictable, on-demand signing may
   enable a chosen-plaintext attack on a zone's private keys. Zones
   using this approach should attempt to use cryptographic algorithms
   that are resistant to chosen-plaintext attacks. It is worth noting
   that although DNSSEC has a "mandatory to implement" algorithm, that
   is a requirement on resolvers and validators -- there is no
   requirement that a zone be signed with any given algorithm.

   The success of using minimally covering NSEC records to prevent zone
   walking depends greatly on the quality of the epsilon functions

chosen. An increment function that chooses a name obviously derived
from the next instantiated name may be easily reverse engineered,
destroying the value of this technique. An increment function that
always returns a name close to the next instantiated name is likewise
a poor choice. Good choices of epsilon functions are the ones that
produce the immediately following and preceding names, respectively,
though zone administrators may wish to use less perfect functions
that return more human-friendly names than the functions described in
Section 4 above.

Another obvious but misguided concern is the danger from synthesized
NSEC records being replayed. It is possible for an attacker to
replay an old but still validly signed NSEC record after a new name
has been added in the span covered by that NSEC, incorrectly proving
that there is no record at that name. This danger exists with DNSSEC
as defined in [3]. The techniques described here actually decrease
the danger, since the span covered by any NSEC record is smaller than
before. Choosing better epsilon functions will further reduce this
danger.

6. Acknowledgements

   Many individuals contributed to this design. They include, in
   addition to the authors of this document, Olaf Kolkman, Ed Lewis,
   Peter Koch, Matt Larson, David Blacka, Suzanne Woolf, Jaap Akkerhuis,
   Jakob Schlyter, Bill Manning, and Joao Damas.

   In addition, the editors would like to thank Ed Lewis, Scott Rose,
   and David Blacka for their careful review of the document.

7. Normative References

   [1] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose,
        "DNS Security Introduction and Requirements", RFC 4033, March
        2005.

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

   [3] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose,
        "Protocol Modifications for the DNS Security Extensions", RFC
        4035, March 2005.

   [4] Bradner, S., "Key words for use in RFCs to Indicate Requirement
        Levels", BCP 14, RFC 2119, March 1997.

Authors' Addresses

   Samuel Weiler
   SPARTA, Inc.
   7075 Samuel Morse Drive
   Columbia, Maryland 21046
   US

   EMail: weiler@tislabs.com

   Johan Ihren
   Autonomica AB
   Bellmansgatan 30
   Stockholm SE-118 47
   Sweden

   EMail: johani@autonomica.se

Full Copyright Statement

Intellectual Property

Acknowledgement