NSEC2 v. DNSNR/NSEC3

Sam Weiler
SPARTA

IETF-60
San Diego
2 August 2004

Outline

- Same
- Different
- Same
- Summary

Outline

- Same (general idea)
- Different (specifics)
- Same (problems)
- Summary

The Old World (NSEC chains)

```
example.com      NS b.example.com.
                 RRSIG(NS)
                 MX ....
                 RRSIG(MX)
                 DNSKEY ....
                 RRSIG(DNSKEY)
                 NSEC a.example.com (NS RRSIG NSEC
                                     DNSKEY MX)
                 RSIG(NSEC)
a.example.com    NS ns1.akamai.com.
                 NSEC b.example.com  (NS NSEC RRSIG)
                 RRSIG(NSEC)
b.example.com A 10.0.0.1
                 RRSIG(A)
                 NSEC example.com (A RRSIG NSEC)
                 RRSIG(NSEC)
```

Hashed Owner Names

```
example.com      NS b.example.com.
  --->a45abd     RRSIG(NS)
                 MX ....
                 RRSIG(MX)
                 DNSKEY ....
                 RRSIG(DNSKEY)
                 NSEC a.example.com (NS RRSIG NSEC
                                     DNSKEY MX)
                 RSIG(NSEC)
a.example.com    NS ns1.akamai.com.
  --->d8edf1     NSEC b.example.com  (NS NSEC RRSIG)
                 RRSIG(NSEC)
b.example.com A 10.0.0.1
  --->45cd25     RRSIG(A)
                 NSEC example.com (A RRSIG NSEC)
                 RRSIG(NSEC)
```

Hashed Owner Names

example.com NS b.example.com.
---->a45abd RRSIG(NS)
MX ....
RRSIG(MX)
DNSKEY ....
RRSIG(DNSKEY)
~~NSEC a.example.com (NS RRSIG NSEC DNSKEY MX)~~
~~RSIG(NSEC)~~
a.example.com NS ns1.akamai.com.
---->d8edf1 ~~NSEC b.example.com (NS NSEC RRSIG)~~
~~RRSIG(NSEC)~~
b.example.com A 10.0.0.1
---->45cd25 RRSIG(A)
~~NSEC example.com (A RRSIG NSEC)~~
~~RRSIG(NSEC)~~
45cd25.example.com NSEC4 a45abd.example.com (NS NSEC4 RRSIG)
RRSIG(NSEC4)
a45abd.example.com NSEC4 d8edf1.example.com (NS RRSIG NSEC4
DNSKEY MX)
RRSIG(NSEC4)
d8edf1.example.com NSEC4 45cd25.example.com (A RRSIG NSEC4)
RRSIG(NSEC4)

Canonical ordering discarded EXCEPT for NSEC records

The New World (Hashed Owner Names)

example.com NS b.example.com.
RRSIG(NS)
MX ....
RRSIG(MX)
DNSKEY ....
RRSIG(DNSKEY)
a.example.com NS ns1.akamai.com.
b.example.com A 10.0.0.1
RRSIG(A)

45cd25.example.com NSEC4 a45abd.example.com (NS NSEC RRSIG)
RRSIG(NSEC4)
a45abd.example.com NSEC4 d8edf1.example.com (NS RRSIG NSEC4
DNSKEY MX)
RRSIG(NSEC4)
d8edf1.example.com NSEC4 45cd25.example.com (A RRSIG NSEC4)
RRSIG(NSEC4)

More similarities

- Choice of hash and salt
- Canonical ordering applies ONLY to NSECx's
  - Apex's NSECx is identifiable, but may be anywhere in the NSECx chain
  - Actual names covered by a given NSECx vary depending on hash and salt
- NSECx's don't have NSECx's
- NSECx owner names and RDATA grow

Big Difference: What gets hashed

- NSEC2 (Ben Laurie)
  - a.b.example.com ---> hash of (a.b.example.com)
    - ef2235cdef2.example.com
  - EXIST RR proves existence of empty non-terminals
    - Exposes names of empty non-terminals
- DNSNR/NSEC3 (Roy Arends)
  - a.b.example.com ---> hash of (a).hash of (b)
    - 34adef231.45edfae341.example.com
  - Exposes structure, not names of empty non-terminals
  - Doesn't work for deep zones (e.g. ENUM)

Other Differences

- NSEC2 (Ben Laurie)
  - Hash iterations
- DNSNR/NSEC3 (Roy Arends)
  - Authoritative-only bit (a.k.a. opt-in)
  - Defines a null hash function (plaintext names)

Other Differences

- NSEC2 (Ben Laurie)
  - Hash iterations
- DNSNR/NSEC3 (Roy Arends)
  - Authoritative-only bit (a.k.a. opt-in)
  - Defines a null hash function (plaintext names)

- All of these are compatible with both proposals
  - We can pick and choose

Outline

- Same (general idea)
- Different (specifics)
- Same (problems)
- Summary

Hash Collisions

a.example.com NS ns1.akamai.com.
NSEC ..... NS NSEC RRSIG
RRSIG(NSEC)

b.example.com A 10.0.0.1
RRSIG(A)
NSEC .... A RRSIG NSEC
RRSIG(NSEC)

What happens when two names hash to the same value?

a.example.com ---> 4fede5623.example.com

b.example.com ---> 4fede5623.example.com

Hash Collisions

a.example.com NS ns1.akamai.com.

b.example.com A 10.0.0.1
RRSIG(A)

a.example.com ---> 4fede5623.example.com
b.example.com ---> 4fede5623.example.com

Two NSEC4's with same name?

4fede5623.example.com NSEC ... (NS RRSIG NSEC)
4fede5623.example.com NSEC ... (A RRSIG NSEC)

Can be used to spoof each other's data away

Hash Collisions

a.example.com NS ns1.akamai.com.

b.example.com A 10.0.0.1
RRSIG(A)

a.example.com ---> 4fede5623.example.com
b.example.com ---> 4fede5623.example.com

Two NSEC4's with same name?

4fede5623.example.com NSEC ... (NS RRSIG NSEC)
4fede5623.example.com NSEC ... (A RRSIG NSEC)
Can be used to spoof each other's data away

Superset of types?

4fede5623.example.com NSEC ... (NS A RRSIG NSEC)
No way to prove the A record doesn't exist
Q: a.example.com IN A?

Change hash algorithm or salt?

Changing Hashes or Salts

- No proposal for doing it incrementally
- Default: Roll through insecure
  - Go completely insecure (remove any DS's in parent and preconfigured SEP keys)
  - Wait for timeout
  - Change hash/salt and resign
  - Go secure (publish DS's; configure SEP keys)

Recursion problem?: NSEC proven not to exist

45cd25.example.com NSEC4 a45abd.example.com (NS...)
a45abd.example.com NSEC4 d8edf1.example.com (NS ...)
d8edf1.example.com NSEC4 45cd25.example.com (A RRSIG ...)

What NSEC covers d8edf1.example.com.?
d8edf1.example.com.--->5e234a

Recursion problem?: NSEC proven not to exist

45cd25.example.com NSEC4 a45abd.example.com (NS...)
a45abd.example.com NSEC4 d8edf1.example.com (NS ...)
d8edf1.example.com NSEC4 45cd25.example.com (A RRSIG ...)

What NSEC covers d8edf1.example.com.?
d8edf1.example.com.--->5e234a

45cd25... PROVES THAT

d8edf1.example.com (hashes to 5e234a)

DOES NOT EXIST

Summary: Differences

- Name covered by the hash
  - Implications for wildcard processing
  - Hashing per-label limits applicability
- Hash iterations (NSEC2)
- "Authoritative-only" bit (a.k.a. opt-in) (NSEC3)
- Option for a non-hashed namespace (NSEC3)

Summary: Problems

- Ambiguous/inconsistent docs
- Wildcard processing not well understood
  - Broken example in NSEC2, no examples in NSEC3
- What to do with collisions
- No (good) way to roll salt/hash
- Possible recursion problem

31 May 2001, draft-ietf-dnsext-delegation-signer-00.txt

19 Mar 2002, DS code circulating privately
28 Mar 2002, DS WGLC concludes (-06)

20 Jun 2002, DS support in a public BIND snapshot
29 Aug 2002, DS lameness bug found
? Oct 2002, grandparent problem found
31 Dec 2002, "Jakob's bug": any NXT is a negative answer
27 Feb 2003, type code roll draft appears

29 May 2003, type code roll into WGLC (-01)

18 Dec 2003, RFC3658 appears (-15)

27 May 2004, RFC3755 appears (-06)

21 Mar 2001, DNSSECbis editors (Scott and Dan) conscripted at IETF-50. WGLC predicted for July 2001

31 May 2001, draft-ietf-dnsext-delegation-signer-00.txt

19 Mar 2002, DS code circulating privately

28 Mar 2002, DS WGLC concludes (-06)

20 Jun 2002, DS support in a public BIND snapshot

29 Aug 2002, DS lameness bug found

? Oct 2002, grandparent problem found

31 Dec 2002, "Jakob's bug": any NXT is a negative answer

27 Feb 2003, type code roll draft appears

29 May 2003, type code roll into WGLC (-01)

18 Dec 2003, RFC3658 appears (-15)

26 Jan 2004, 1st WGLC on DNSSECbis concludes

27 May 2004, RFC3755 appears (-06)

2 June 2004, 2nd WGLC on DNSSECbis concludes

20 July 2004, DNSSECbis advanced to IESG (-11, -07, -09)