draft-vcelak-nsec5-04

NSEC5: DNSSEC Authenticated Denial of Existence

Jan Vcelak (CZ.NIC)
Sharon Goldberg (Boston University)
Dimitrios Papadopoulos (University of Maryland)
Shumon Huque (Salesforce)
David C. Lawrence (Akamai)
## DNSSEC Authenticated Denial of Existence

<table>
<thead>
<tr>
<th></th>
<th>No offline zone enumeration</th>
<th>Integrity vs outsiders</th>
<th>Integrity vs compromised nameserver</th>
<th>No online crypto</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS (legacy)</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>NSEC or NSEC3</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Online Signing (“NSEC3 White Lies”)</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NSEC5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
</tr>
</tbody>
</table>

**NSEC5** replaces **SHA1** used in NSEC3 with a Verifiable Random Function (VRF) [draft-goldebe-vrf-00] that resolvers cannot compute offline.
offline signing with NSEC5

\[ H(\Pi_{a.com}) = 9ae3e \]

“Hash” with secret VRF key

a.com
c.com
z.com
offline signing with NSEC5

\[ H(\Pi(a.com)) = 9ae3e \]
\[ H(\Pi(c.com)) = 8cb67 \]
\[ H(\Pi(z.com)) = 3cd91 \]
offline signing with NSEC5

\[ H(\Pi_{a.com}) = 9ae3e \]
\[ H(\Pi_{c.com}) = 8cb67 \]
\[ H(\Pi_{z.com}) = 3cd91 \]

sort

3cd91
8cb67
9ae3e

“Hash” with secret VRF key

Sign NSEC5 records with secret ZSK

3cd91.com
8cb67.com
9ae3e.com
offline signing with NSEC5

VRF hash

$H(P(a.com)) = 9ae3e$

VRF proof

[Think of this as a deterministic public-key signature]

"Hash" with secret VRF key

a.com

c.com

z.com

* NSEC5-ECC: VRF based on elliptic curves
  • [draft-goldbe-vrf-00]. (Presented at SAAG, Thursday!)
  • Has a formal cryptographic security proof.
  • For 256-bit elliptic curves, $\Pi$ gives 641-bit outputs.
offline signing with NSEC5

\[ \text{Hash} \text{ with secret VRF key} \]

\[ \text{Sign NSEC5 records with secret ZSK} \]
answering queries with NSEC5

Public Zone Signing Key (ZSK):

Public VRF Key:

q.com?

H(aa867) = 7a89b

Π(q.com) = aa8678

secret VRF key

a.com

3cd91.com

8cb67.com

8cb67.com

9ae3e.com

9ae3e.com

3cd91.com
answering queries with NSEC5

Public Zone Signing Key (ZSK):

Public VRF Key:

q.com?

3cd91.com
8cb67.com

\[ \Pi(q.com) = \text{aa8678} \]

\[ H(\text{aa867}) = 7a89b \]

secret VRF key

\[
\begin{align*}
3cd91.com \\
8cb67.com \\
9ae3e.com \\
9ae3e.com \\
3cd91.com
\end{align*}
\]
answering queries with NSEC5

Public Zone Signing Key (ZSK):

Public VRF Key:

q.com?

3cd91.com
8cb67.com

PROOF
aa8678

7a89b

\( H(aa867) = 7a89b \)

\( \Pi(q.com) = aa8678 \)

secret VRF key

\[
\begin{align*}
3cd91.com \\
8cb67.com \\
9ae3e.com
\end{align*}
\]
answering queries with NSEC5

Public Zone Signing Key (ZSK):

Public VRF Key:

$q$ com?

PROOF

aa8678

3cd91.com

8cb67.com

secret VRF key

$H(aa867) = 7a89b$

$\Pi(q.com) = aa8678$

To verify:

Does NSEC5 cover PROOF?

$3cd19 < H(aa8678) < 8cb67$

Does PROOF match query?

VER $(q.com, aa8678)$

[Think of this as a signature verification]
## DNSSEC Authenticated Denial of Existence

<table>
<thead>
<tr>
<th></th>
<th>No offline zone enumeration</th>
<th>Integrity vs outsiders</th>
<th>Integrity vs compromised nameserver</th>
<th>No online crypto</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DNS (legacy)</strong></td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td><strong>NSEC or NSEC3</strong></td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td><strong>Online Signing</strong> (“NSEC3 White Lies”)</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td><strong>NSEC5</strong></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
</tr>
</tbody>
</table>

- **Because resolvers cannot compute VRF hashes offline**
- **Because the nameserver doesn’t know the zone-signing key**
- **In [NDSS’15] we proved this is necessary to prevent zone enumeration & have integrity**
Knot DNS & Unbound
authoritative nameserver & recursive resolver

-04 draft includes DNS-level optimizations:
  1. The wildcard bit from [draft-gieben-nsec4-00]
  2. Precomputed closest encloser proofs mentioned in [RFC7128]

9K Lines of Code, no new libraries (openSSL) or system optimizations

Current implementations support P-256 curve.
Could be faster with Ed25519 curve included in the -04 draft
empirical measurement of NXDOMAIN response sizes

response size (bytes)

Ethernet MTU: 1500 bytes

- NSEC3 RSA-2048
- NSEC3 ECDSA-P256
- NSEC5 ECC-P256
- NSEC3 RSA-1024 “legacy”
nameserver query throughput (steady rate, NXDOMAIN)

zone: an Alexa-100 SLD

throughput

query rate

Machine specs: 20X Intel(R) Xeon(R) CPU E5-2660 v3 @ 2.60GHz Dual Mode (Total 24 threads on 40 virtual CPUs) 256GB RAM running CentOS Linux 7.1
questions?

• Research paper with performance numbers & crypto proofs:
  http://ia.cr/2017/099
• NSEC5 Project page
  https://www.cs.bu.edu/~goldbe/papers/nsec5.html
• Long preso on NSEC5 at Real World Crypto (RWC’17)
  https://www.youtube.com/watch?v=-pWrij0YhGo

dnsreactions…

Hearing about NSEC5

When I finally grasp NSEC5
backup slides
offline zone signing with NSEC3 [RFC5155]

\[ \text{SHA1}(a.com) = \text{a1bb5} \]
\[ \text{SHA1}(c.com) = \text{23ced} \]
\[ \text{SHA1}(z.com) = \text{dde45} \]

Hash names

23ced
a1bb5
dde45

Sign NSEC3 records with secret ZSK

23ced.com
a1bb5.com
dde45.com

a.com
c.com
z.com
answering queries with NSEC3

Public Zone Signing Key (ZSK): key icon

SHA1(q.com) = c987b

Step 1: Collect
a1bb5.com
dde45.com

q.com?

a.com
z.com

dde45.com

23ced.com
a1bb5.com
dde45.com

23ced.com
offline zone enumeration with NSEC3

Public Zone Signing Key (ZSK):

SHA1(r.com) = 33c46

Step 1: Collect
a1bb5.com
don45.com

Step 2: Crack
a.com
c.com
z.com

doen45.com
23ced.com

Offline dictionary attack
online signing stops zone enumeration!

Public Zone Signing Key (ZSK):

SHA1(r.com) = 33c46

r.com?

33c45.com
33c47.com

a.com
c.com
z.com

secret ZSK

“NSEC3 White Lies” [RFC7128]