# NSEC5 and Zone Enumeration

Johan Lanzrein

March 2019

**Abstract**

This paper discusses a new approach to DNSSEC. With the current implementations of DNSSEC, we face the problem of zone enumeration. An attacker can make easily mount an attack where he guesses all the domain names present in a zone. This is a privacy flaw that should be addressed. In the paper, researchers give a formal proof as to why the current implementation of DNSSEC, called NSEC and NSEC3, suffer from zone enumeration. Afterwards, they build a new protocol, NSEC5, which is robust against zone enumeration while keeping the same advantages of NSEC3. The main idea is to use an online public key cryptographic operation.

## 1 Introduction

### 1.1 DNS

As an introduction to the subject, we recall what DNS is. DNS stands for Domain Name System. It is an internet system meant to help clients connect with servers. The DNS is mainly decentralized as there are many server thorough the world that serve requests from around the world. A client is usually an internet web-browser. For a more detailed and rigorous explanation on DNS, see [3].

### 1.2 Attacks on DNS

DNS being a protocol that relies on the internet to relay queries, is weak to any type of man-in-the-middle attacks. For example, a malicious user $M$ could easily spoof the local DNS server and reply to the requests of the client to redirect them towards a fake website.
Therefore, research has been done on how to communicate DNS messages over the internet in a secure way. The first project was NSEC, followed shortly after by NSEC3. However both of these approaches, while protecting against man-in-the-middle attacks, opened the way to a new type of attacks, i.e. zone enumeration.

## 2 NSEC and zone enumeration

### 2.1 DNSSEC

DNSSEC is a standard proposed to offer authenticated positive response to DNS queries and authenticated denial-of-existence. To authenticate positive responses, a trusted server (also referred to as a Primary server) will sign the values in the zone and transmit them to a non-trusted server who serves as intermediary (called a Secondary server). However, to authenticate denial-of-existence it is more complicated. This is the goal of NSEC.

### 2.2 NSEC

We will refer to NSEC and NSEC3 indifferently with the same name: NSEC, as both of them are very similar. However, we will point out the differences when needed.
Both algorithms rely on the same idea.

**Algorithm 1** NSEC Setup

```
1: $L \leftarrow \emptyset$
2: if NSEC then
3:     $L \leftarrow R$
4: else if NSEC3 then
5:     for $x \in R$ do
6:         $L \leftarrow L \bigcup h(x)$
7: sort $L$ in lexicographical order
8: $\Sigma = \{Sign_{SK}(y_j, y_{j+1})$ for all $y_j \in L\}$
9: return $\Sigma$
```

## 2.3 Zone enumeration

The NSEC and NSEC3 approach stay very weak, because an attacker could mount an attack by repeatedly querying for non-existent domains and getting the entire list $\Sigma$. Afterwards, if NSEC is used, the attacker has access to all the domain names. And if NSEC3 is used, the attacker can use rainbow tables or dictionary attacks to break the hashes.
This type of attacks are zone enumeration attacks. While the discussion on whether or not it was a problem is a whole other argument, we will here assume it is a problem for privacy. Moreover an attacker could use zone enumeration to mount other attacks such as spamming or use the names as possible emails for spamming.
An attacker could also use the whois database to find information on the people in the zone.

## 3 PSR Model

The PSR model is an interactive proof system consisting of three parties : primary, secondary, resolver (hence the PSR initials). The goal of this model is to have a secure denial-of-existence proof and no zone enumeration.

### 3.1 Algorithms in PSR

Let $U$ be a universe of element, $V$ be a set of possible values. PSR consists of the four following algorithms :

- $Setup(R, v, 1^k) \rightarrow (PK, I_s)$:
  - Input: $R$ is a set of names in the zone, $v$ is the mapping of the names $R$ to their IP addresses $V$, $1^k$ is the security parameter.
  - Output: $PK$ are the public parameters given to both the resolvers and the secondaries, $I_s$ is information given only to the secondaries.
- $Query(x, PK) \rightarrow (q)$:
  - Input: $x$ is the name to query, $PK$ public parameters.
  - Output: $q$ is the output query sent to the secondary.
- $Answer(q, I_s, PK) \rightarrow (b, v, \pi)$:
  - Input: $q$ the query, $I_s$ the information from primary, $PK$ public parameters.
  - Output: $b$ a bit, $v$ mapping of $x$ if it exists, $\pi$ proof for the value.
- $Verify(b, v, \pi) \rightarrow (g)$:
  - Input: $(b, v, \pi)$ output of $Answer$.
  - Output: $g$ a bit 1 if the answer is valid, 0 else.

## 4 Formalism

Let us now describe a few definitions that are used to explain why the PSR system is protected against zone enumeration.

## 4.1 Completeness

A PSR system should satisfy the completeness property. This property states that when all members follow the protocol exactly, the output of $Verify$ should be 1 with probability $1-\mu(k)$. $\mu(k)$ being a negligible function of k.

## 4.2 Soundness

Soundness states that even if a malicious adversary takes over the secondary, then it is not possible to convince a resolver of a false response. This must be satisfied with probability $\mu(k)$, $\mu(k)$ a negligible function. Mathematically, we have : $Verify(b, v, \pi) = 1 \wedge ((x \in R \wedge (b = 'no' \vee v \neq v(x)) \vee (x \notin R \wedge b = 'yes'))$.

### 4.2.1 Privacy

First let us discuss what security against selective membership is. Informally, this means that an adversary can not infer information about elements in the set $R$ by querying repeatedly the $Answer$ algorithm. Meaning by querying $\{y_1, ..., y_r\}$, he can't decide if $x \in R$ for some $x \notin \{y_1, ..., y_r\}$. Notice that this is close to the property of privacy we would like to have.
PSR system however satisfies an even stronger property called f-zero-knowledge property. A PSR system is said to be f-zero knowledge if there is a simulator that can simulate the behavior of the PSR for which no adversary A and no distinguisher D can tell the difference between the simulator and the PSR system better than randomly guessing with a negligible advantage $\mu(k)$. We can see that the definition of f-zero knowledge is very strong therefore there is a reduction from f-zero knowledge to security against selective membership. If the system is secure against selective membership, we can convince ourselves that it is protected against zone enumeration.

**Theorem 1** *PSR as a f-zk* $(Setup, Query, Answer, Verify)$ *constitute a f-zk PSR for the function* $f(R) = |R|$

The theorem at 4.2.1 is proved by successively proving the property of completeness, soundness and privacy. We won't go into the detail of the proof however it can be found as Theorem III.1 in [1].

## 5 NSEC5 algorithms

As NSEC5 is a PSR system, there will be four algorithms $(Setup, Query, Answer, Verify)$. It relies on one online RSA computation and two hash functions and a signature scheme:

- RSA keys : $(PK_s, SK_s) = ((e_s, N_s), (d_s, N_s))$.
- $h_1 : U \to \{0,1\}^{|N_s - 1|}$.
- $h_2 : \mathbb{Z}_{N_s} \to \{0,1\}^n$.
- Keys for a signature scheme $Sign_{SK_p}, Ver_{PK_p}$ : $(PK_p, SK_p)$.

The algorithms are given on the next page.

## References

[1] S. Goldberg, M. Naor, D. Papadopoulos, L. Reyzin, S. Vasant, and A. Ziv. Nsec5: Provably preventing dnssec zone enumeration. Cryptology ePrint Archive, Report 2014/582, 2014. `https://eprint.iacr.org/2014/582`.

[2] B. Laurie, G. Sisson, R. Arends, and D. Blacka. Dns security (dnssec) hashed authenticated denial of existence. RFC 5155, RFC Editor, March 2008. `http://www.rfc-editor.org/rfc/rfc5155.txt`.

[3] P. Mockapetris. Domain names - concepts and facilities. STD 13, RFC Editor, November 1987. `http://www.rfc-editor.org/rfc/rfc1034.txt`.

**Algorithm 2** NSEC5 Setup$(R, v, 1^k)$

```
1: Choose hash function $h_1, h_2$
2: Generate key pair $(PK_p, SK_p)$ for $Sign$
3: Generate key pair $(PK_s, SK_s)$ for RSA permutation
4: $PK \leftarrow (PK_p, PK_s, h_1, h_2)$
5: $\Sigma \leftarrow \emptyset$
6: $L \leftarrow \emptyset$
7: for all $x \in R$ do
8:     $\Sigma \leftarrow \Sigma \bigcup Sign(x, v(x))$
9: for all $x \in R$ do
10:    $\pi \leftarrow (h_1(x))^{d_s} mod\ N_s$
11:    $y \leftarrow h_2(\pi)$
12:    $L \leftarrow L \bigcup y$
13: Sort $L$ in lexicographical order
14: $\Pi = \{Sign_{SK_p}(y_j, y_{j+1})$ for all $y_j \in L\}$
15: $I_s \leftarrow (SK_s, \Sigma, \Pi)$
16: return $(PK, I_s)$
```

**Algorithm 3** NSEC5 Query$(x, PK)$

```
1: Compute a query $q$ from $x$
2: return $q$
```

**Algorithm 4** NSEC5 Answer$(q, I_s, PK)$

```
1: if $x \in R$ then
2:     $\pi \leftarrow Sign_{SK_p}(q, v(q))$
3:     $v \leftarrow (q, v(q))$
4:     return $['yes', v, \pi]$
5: if $x \notin R$ then
6:     $\pi_y \leftarrow h_1(q)^{d_s} mod\ N_s$
7:     $y \leftarrow h_2(\pi_y)$
8:     In $\Pi$ find two closest $(y_j, y_{j+1})$ such that $y_j < y < y_{j+1}$
9:     $u \leftarrow Sign_{SK_p}(y_j, y_{j+1})$
10:    $\pi \leftarrow (y_j, y_{j+1}, (\pi_y, u))$
11:    return $['no', \perp, \pi]$
```

**Algorithm 5** NSEC5 Verify$(b, v, \pi)$

```
1: if $b = 'yes'$ then
2:     return $Ver_{PK_p}(\pi)$
3: if $b = 'no'$ then
4:     $(y_j, y_{j+1}, (\pi_y, u)) \leftarrow \pi$
5:     Verify $Ver_{PK_p}(u) = 1$
6:     Verify $y_j < h_2(\pi_y) < y_{j+1}$
7:     Verify $h_1(q) = \pi_y^{e_s} mod\ N_s$
8:     return if all three pass 1 else 0
```