

# P00011 - Research and Study Methods

## Improving DNS security by relying on DNSSEC

Arnaud Fontaine (08090091)

Project proposer: Faye Mitchell

19th November 2008

**Subject:** *With the major DNS flaw that was discovered earlier this year there has been a push to implement DNSSEC. Discuss DNSSEC and how it improves the security of the DNS system to a variety of attacks, including the major flaw discovered earlier this year.*

## **Abstract**

Nowadays, DNS is currently a fundamental part of Internet infrastructure as it provides translation between an IP address and a domain name and thus is used everywhere. However, the major DNS protocol security flaw discovered this year by Dan Kaminsky along with the existing ones raise important concerns about the future of DNS from a security point of view and how to make it viable in a hostile environment such as Internet.

This paper firstly gives a brief overview of DNS protocol before outlining current DNS protocol security flaws. Then, it describes DNSSEC and how the deployment of these DNS extensions could be able to resolve the major security issues of DNS.

Main DNS security issues involve cache poisoning and spoofing which may lead for instance to thievery of important information. DNSSEC provides a complete solution by the concept of chains of trust, unfortunately it is not yet widely used.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>DNS overview</b>	<b>1</b>
<b>3</b>	<b>DNS security threats</b>	<b>3</b>
3.1	ID guessing and query prediction . . . . .	4
3.2	Name chaining and cache poisoning . . . . .	6
<b>4</b>	<b>DNSSEC</b>	<b>7</b>
<b>5</b>	<b>Conclusion</b>	<b>10</b>

# List of Figures

1	<i>DNS recursive query example</i> . . . . .	3
2	<i>Abstract view of DNS queries</i> . . . . .	4
3	<i>DNS recursive query with DNSSEC</i> . . . . .	9

# List of Tables

1	Main DNS Resource Records (RRs) fields . . . . .	2
2	New RRs introduced by DNSSEC . . . . .	8

# Nomenclature

DoS	Denial of Service
ISP	Internet Service Provider
RFC	Request For Comments
TLD	Top Level Domain
TTL	Time To Live

# 1 Introduction

Nowadays, DNS is widely used as protocol allowing translation between hard-to-remember IP addresses and easy-to-remember domain names, thus making it one of the most important foundation part of Internet and more generally networks. However, DNS was designed a long time ago without security in mind, especially because it was not supposed to be scaled this way at that time. DNS has several important security flaws that DNSSEC is currently supposed to address. However, how is it supposed to address all these security issues? Is DNSSEC a viable possibility and especially what is the current deployment status?

The first part of this paper introduces main DNS protocol concepts necessary to understand important DNS concepts and shortcomings. Secondly, it discusses about security flaws discovered along with description of the last serious DNS flaw announced this year. Finally, it provides an overview of DNSSEC which aims to address these issues and what is its current deployment status.

## 2 DNS overview

In this section, I will introduce briefly DNS protocol<sup>1</sup> to give necessary concepts to understand attacks against this protocol and DNSSEC.

DNS provides a scalable and simple way to map a domain name<sup>2</sup> and a hard-to-remember IP address as a hierarchical tree from root DNS servers (namely the final . in canonical representation) to sub-domains (for instance *test.example* whose canonical representation would be *test.example.org*. and managed by *example.org* DNS server). Each node or level within a hierarchy is called a DNS zone.

---

<sup>1</sup>Mockapetris (1987a), Mockapetris (1987b), Wikipedia (August 2001) and Tanenbaum (August 2002, Section 7.1 )

<sup>2</sup>A domain name may be relative (*test* is relative to *example.org*.) or absolute (ends with a dot such as *test.example.org*).

More precisely, DNS intends to map domain names onto resource records (RRs). The table 1 summarizes main RRs and their meaning (their types are not given in this document but may be found in Mockapetris (1987b)).

Table 1: Main DNS Resource Records (RRs) fields

Field	Description
NAME	Domain name to which this RR apply.
TYPE	RR record type (SOA, A, CNAME...)
CLASS	Usually IN for Internet.
TTL	Specify time interval before this RR (which may be in cache on another DNS server) is checked again.
RDATA	Resource description dependent of CLASS and TYPE

Commonly, DNS relies on UDP transport protocol and server daemon uses port 53. DNS distinguishes an authoritative record, coming from the DNS server that actually owns the RR, and a cache record. Once an user queries a domain name server, it will be stored into a cache during a TTL interval. The figure 1 shows the query process from an user A with a . b . c . d IPv4 address who wants to know the IPv4 address associated with *test.example.org.*, this process is usually referred as a recursive query (in this example, we assume that the ISP recursive name server<sup>3</sup> doesn't already have the RR for *test.example.org.*).

Another important point about DNS protocol is the TID (*Transaction ID*) field which is assigned by any program generating a DNS query. Usually, the tuple of the TID (16-bits long) and the port number (16-bits long) uniquely identifies a query and its reply.

The figure 2 is an abstract view of the DNS architecture, only considering the following (the letters on the figure is for later references):

---

<sup>3</sup>A recursive name server is a name server which allows to ask to another server because it is not authoritative for the zone

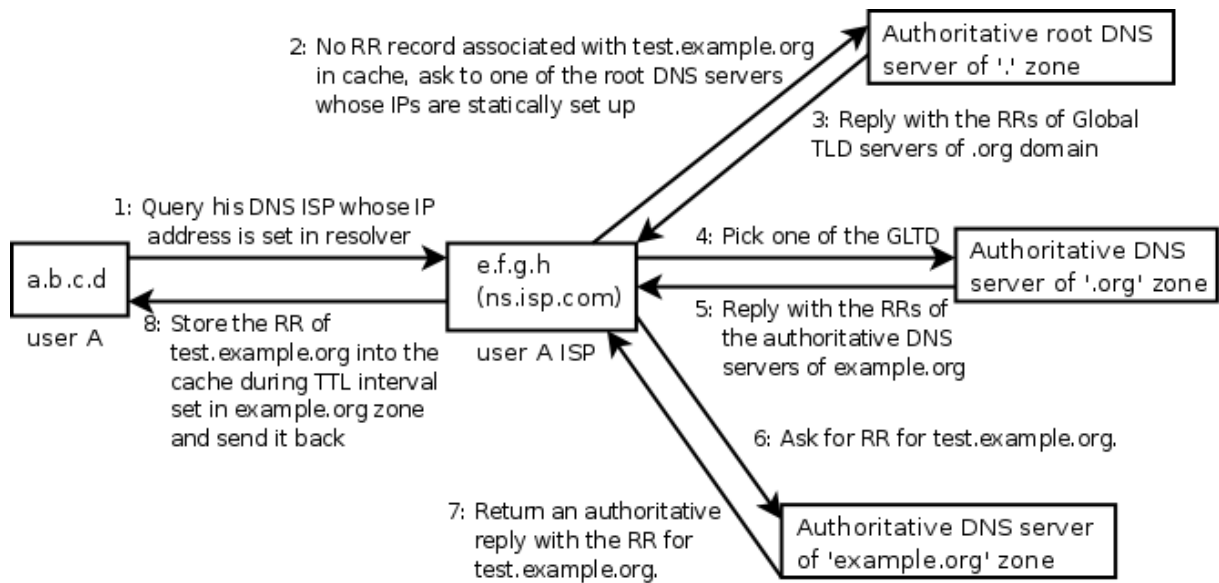


Figure 1: *DNS recursive query example*

- A master and slave authoritative DNS server on their zones (indeed the slave is just a replication of the master for redundancy purpose or for load-balancing), there may have more than one slave.
- A forwarder (also known as a recursive name server) which may also caches the RR forwarded (for instance, on figure 1, the ISP name server is a forwarder of the query sent by user A), there may have more than one forwarder.
- A resolver which is the end-point client requesting a RR for a given IP or domain name.

### 3 DNS security threats

DNS protocol was designed in the 1980s, however at that time security was not considered as important as it is now, mainly because it was not so popular and widely used. Therefore, DNS protocol contains many security issues as stated in Atkins & Austein (August 2004).

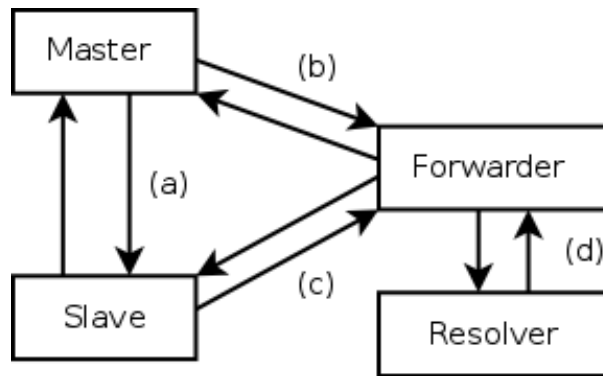


Figure 2: *Abstract view of DNS queries*

However, I think that some issues given in this paper are not tightly relevant to the DNS protocol:

- **DoS**

It may be performed against a DNS server but in my opinion it should be handled by the lower-layer protocols as it is common to all application-level protocols.

- **Betrayal By Trusted Server**

A trusted server sends back voluntarily or unintentionally the wrong RR for a query from a resolver ((a), (b) and (c) on figure 2). I think it is not really a flaw within DNS protocol but is more a matter of trust because it is indeed common to all application-level protocols like DOS.

The attacks described in the following sections are more relevant to the DNS protocol and are easier to perform as DNS uses UDP for queries (no handshake like TCP protocol does, but it would involve a huge overhead concerning DNS).

### 3.1 ID guessing and query prediction

As described in section 2, a reply matches up a query previously sent according to the port number and the TID. It is worth noticing that this attack may only take

place if the attacker knows the query `TYPE` and `NAME` or get it by intercepting packets as described below). Therefore, if an attacker wants to spoof the reply sent back to the resolver from a forwarder ((*d*) on figure 2), it will have to provide valid information for the following fields: resolver port number and TID (the server port number is well-known, e.g. 53).

The attacker may decide to proceed using one of the following methods to get these information:

- **Packet interception**

Determine the resolver port number and TID by intercepting the query packet initially sent to the forwarder and forge a valid packet (man-in-the-middle). At a first glance, this method looks pretty easy but it can only be performed if the attacker and the resolver are on a shared network or the attacker is on a transit network (namely man-in-the-middle attack).

- **Brute force**

As the port number and the TID are both 16-bits fields, there are  $2^{32}$  combinations of ports and TIDs possible. Thus, the attacker could send  $2^{32}$  different packets, one would inevitably matches up. In addition, as source port number is ephemeral, it is commonly includes within [1024, 65535] range (as an example GNU/LINUX kernel usually defines a range of [32768, 61000] for system with > 128MB of memory), consequently reducing the complexity of the attack.

In early DNS implementations, the TID was calculated sequentially (e.g. it was simply incremented by one for each outgoing request), thus the attacker may perform a query to get the current TID ( $n$ ) before the one from the resolver and then forge packets based on the assumption that the TID starts at  $n + 1$ . Fortunately, nowadays, most DNS implementations calculate a random TID.

As a matter of course, this kind of attack is only feasible if the attacker reply sent to the resolver beats the one from the forwarder, so it reduces the probability

of a successful brute force attack (except if the attacker performs a DOS on the forwarder).

### 3.2 Name chaining and cache poisoning

These attacks are considered as the most interesting as it allows an attacker to inject bogus data into a recursive DNS server cache, so a resolver who will subsequently query this server would get bad information (Tanenbaum August 2002, Section 8.9.2 ) (Wikipedia October 2003) ((*d*) on figure 2). For instance, an attacker may performed this kind of attack to maliciously redirect *test.example.org* to another domain name aiming to get personal information. Dan Kaminsky discovered several variants of this attack (Kaminsky 6th August 2008, Friedl 7th August 2008) that has been widely spread among the media and considered as the most serious vulnerabilities in DNS protocol.

A naming chaining attack aims to inject arbitrary data within a RDATA field (see section 2) which contains a domain name (namely NS, CNAME, DNAME, MX and SRV DNS TYPE fields) in the forwarder cache. A DNS server which does not check carefully the relevancy of the reply to an original query is particularly vulnerable to this threat.

Cache poisoning is considered more important as it allows feeding the victim forwarder cache with bad RRs even if the server is properly implemented and is considered as a flaw of DNS protocol.

Both attacks relies on the persistence of the cache stored by a server, because once a RR has been stored in the cache, it will not be checked until TTL period is reached. It is made possible because UDP does not allow to ensure that the answer is actually supplied by the authoritative server of a zone.

A common cache poisoning attack intending to redirect *test.example.org.* to *a.b.c.d* attacker IP address for resolvers using *ns.isp.com* would be achieved in the following steps (assuming that *test.example.org.* is not already in the cache, otherwise the attacker would have to wait until the TTL expires):

1. The attacker sends a query to *ns.isp.com* recursive name server requesting *test.example.org*. IP address.
2. The query will be carried out along the DNS hierarchy as stated on figure 1. In the meantime, the attacker forges and sends A RR which associates attacker IP address with *test.example.org*. as described in section 3.1.
3. If one of the attacker forged packet matches the TID and arrives before the reply from the authoritative server, then it will be stored in the cache for a TTL interval and the latter reply will be ignored.
4. Subsequent resolver queries for *test.example.org* on *ns.isp.com* will returned *a.b.c.d*.

Dan Kaminsky extends this attack by giving a method to inject into a forwarder cache the authority record, thus allowing to redirect completely a target DNS zone.

## 4 DNSSEC

DNSSEC (Wikipedia February 2005, Eastlake 3rd March 1999) intends to address most of DNS security threats mentioned above while maintaining a backward-compatibility with original DNS protocol, permitting an incremental deployment. As stated by Eastlake 3rd (March 1999), DNSSEC brings the following security features to DNS protocol:

- Data integrity and authentication.
- Public keys distribution.
- Transactions and requests authentication.

Basically, all DNS replies are digitally signed along their path and then checked by the resolver at the end-point, thus ensuring that the data have not been tampered or forged. Furthermore, DNSSEC also allows to publish other types of information requiring authentication or public key distribution, for instance SSHFP (specific RR publishing a SSH public host key fingerprint to verify host authenticity).

DNSSEC relies on asymmetric public key algorithm for the digital signature (uses a public and private keys pair). The table 2 describes the new RRs introduced by DNSSEC along with their meaning as given in DNSSEC RFCs. DNSSEC also adds information to the DNS packets headers. A zone administrator would proceed as below to set up DNSSEC:

Table 2: New RRs introduced by DNSSEC

<b>RR</b>	<b>Description</b>
DNSKEY	DNS public key
RRSIG	RR signature
DS	Delegation signer
NSEC	Next secure

1. Generate a public and private keys.
2. Hold the zone public key, determine the cryptographic algorithm which is going to be used (SHA 1, MD5 for example) minus other information.
3. The RRs are grouped together by sets (RRSets) of NAME, CLASS and TYPE (table 1).
4. Each set is digitally signed by the private key and then stored in RRSIG RR. From now, The private key can be put offline in a secure place once this operation has been done.
5. Set the DS, which is actually a pointer to the next authoritative server below in the hierarchy, and the NSEC RR, which allows authentication of TYPE and NAME non-existence.

Therefore we have the following chain of trust (as shown on figure 3 where ZSK means RRSIG and KSK is the private key, source: Lambert et al. (2008)):

- The data for a given zone are considered correct if they have been previously signed by the private key.

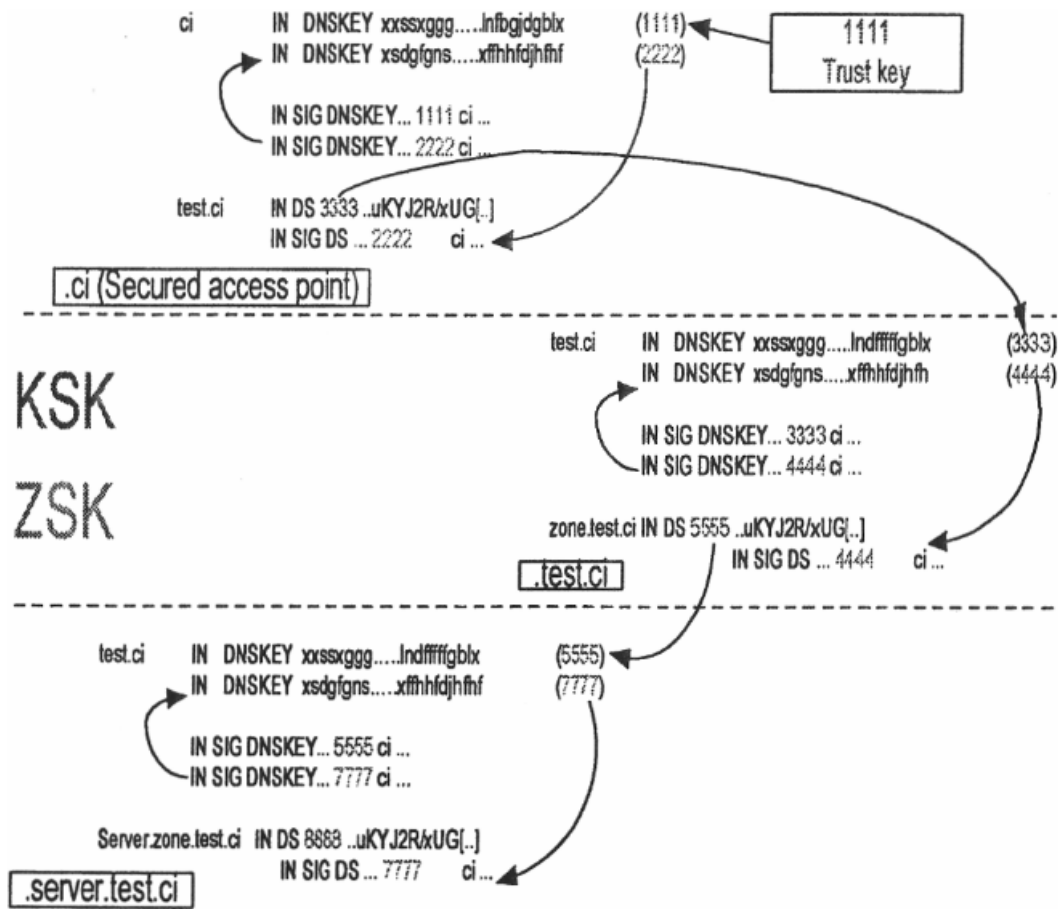


Figure 3: DNS recursive query with DNSSEC

- A RRSIG is valid only if it has been signed by the private key.
- The private key is valid only if referred by a trusted DS RR.
- A DS RR is valid only if it has been signed by the parent RRSIG or the latter has been exchanged offline.

This way, DNSSEC may be viewed as a chain of trust from the root name server to the resolver which will check the digital signature. This chain of trust ensures that forged or tampered packets are considered invalid and also ensures that cache poisoning is not possible anymore. Thus, DNSSEC solves most important security issues related to original DNS protocol flaws even if it does not solve

issue like DOS.

Indeed, for various reasons, the root name server are not signed (except Brazil (.br), Bulgaria (.bg), Czech Republic (.cz), Puerto Rico (.pr) and Sweden (.se)), so current DNSSEC implementations (known as DNSSEC-bis rely on a registry called DLV (DNSSEC *Look-aside Validation*) which acts as a the chain of trust parent, so the DLV is actually queried instead of the root name server.

## 5 Conclusion

The security flaw discovered by Dan Kaminsky is rather serious (even if a fix has been published) and should remain us that DNSSEC is not an option anymore and should be used as soon as possible by using early deployment method such as DLV registry provided by the Internet Systems Consortium. At the moment, only some GTLDs provide complete DNSSEC but the transition to DNSSEC has been planned to finish on 2011.

However, DNSSEC has some downsides: it increases computational cost on the resolver side and it is quite difficult to implement properly. But these (minor) issues shouldn't be a problem in the future when DNSSEC will be widely deployed.

## References

- Atkins, D. & Austein, R. (August 2004), 'Threat Analysis of the Domain Name System (DNS)', RFC 3833 (Informational).  
**URL:** <http://www.ietf.org/rfc/rfc3833.txt>
- Eastlake 3rd, D. (March 1999), 'Domain Name System Security Extensions', RFC 2535 (Proposed Standard).  
**URL:** <http://www.ietf.org/rfc/rfc2535.txt>
- Friedl, S. (7th August 2008), 'An Illustrated Guide to the Kaminsky DNS Vulnerability'. *last access: 09/17/2008*.  
**URL:** <http://www.unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html>
- Kaminsky, D. (6th August 2008), It's The End of Cache As We Know It, in 'Black-Hat 2008', Las Vegas, USA. *last access: 09/17/2008*.  
**URL:** [http://www.doxpara.com/DMK\\_BO2K8.ppt](http://www.doxpara.com/DMK_BO2K8.ppt)
- Lambert, K. T., Souleymane, O., Tiemoman, K., Brice, A. & Pierre, T. (2008), 'Deployment of DNSSEC: Problems and outlines', *Research Challenges in Information Science, 2008. RCIS 2008. Second International Conference on* pp. 343–348.
- Mockapetris, P. (1987a), 'Domain names - concepts and facilities', RFC 1034 (Standard).  
**URL:** <http://www.ietf.org/rfc/rfc1034.txt>
- Mockapetris, P. (1987b), 'Domain names - implementation and specification', RFC 1035 (Standard).  
**URL:** <http://www.ietf.org/rfc/rfc1035.txt>
- Tanenbaum, A. S. (August 2002), *Computer Networks, Fourth Edition*, Prentice Hall, ISBN 0-13-066102-3.
- Wikipedia (August 2001), 'Domain Name System'. *last access: 09/16/2008*.  
**URL:** [http://en.wikipedia.org/wiki/Domain\\_Name\\_System](http://en.wikipedia.org/wiki/Domain_Name_System)
- Wikipedia (February 2005), 'DNS cache poisoning'. *last access: 09/18/2008*.  
**URL:** [http://en.wikipedia.org/wiki/DNS\\_cache\\_poisoning](http://en.wikipedia.org/wiki/DNS_cache_poisoning)

Wikipedia (October 2003), 'Domain Name System Security Extensions'. *last access: 09/16/2008.*

**URL:** *<http://en.wikipedia.org/wiki/DNSSEC>*

**Words count: 2802 (excluding tables and subject title)**