Consequences of compromised zone keys in DNSSEC

Gilles Guette

I R I S A
INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION
INTERNE
N° 1787

CONSEQUENCES OF COMPROMISED ZONE KEYS IN DNSSEC

GILLES GUETTE

ISSN 1166-8687

IRISA
CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES
Campus de Beaulieu – 35042 Rennes Cedex – France
Tél. : (33) 02 99 84 71 00 – Fax : (33) 02 99 84 71 71
http://www.irisa.fr

Consequences of compromised zone keys in DNSSEC

Gilles Guette*

Systèmes communicants
Projet Armor

Publication interne n1787 — Mars 2006 — 13 pages

Abstract: The Domain Name System is a distributed tree-based database. The DNS protocol is largely used to translate a human readable machine name into an IP address. The DNS security extensions (DNSSEC) has been designed to protect the DNS protocol. DNSSEC uses public key cryptography and digital signatures. A secure DNS zone owns at least a key pair (public/private) to provide two security services: data integrity and authentication. To trust some DNS data, a DNS client has to verify the signature of this data with the right zone key. This verification is based on the establishment of a chain of trust between secure zones. To build this chain of trust, a DNSSEC client needs a secure entry point: a zone key configured as trusted in the client. And then, the client must find a secure path from a secure entry point to the queried DNS resource. Zone keys are critical in DNSSEC and are used in every steps of a name resolution. In this report, we present a study on consequences of a compromised key in DNSSEC. We describe compromised key attacks and we present current defenses.

Key-words: DNSSEC, key compromission, network security

(Résumé : tsvp)

* gilles.guette@irisa.fr

Centre National de la Recherche Scientifique
(UMR 6074) Université de Rennes 1 – Insa de Rennes

Institut National de Recherche en Informatique
et en Automatique – unité de recherche de Rennes

Conséquences de clés de zone compromises dans DNSSEC

Résumé : Le sytème de noms de domaine est une base de donnée distribuée basée sur un modèle arborescent. Le protocole DNS est largement utilisé pour effectuer essentiellement la correspondance entre un nom de machine et son adresse IP. Les extensions de sécurité du DNS (DNSSEC) ont été conçues pour protéger ce protocole. Pour cela, DNSSEC utilise la cryptographie à clé publique ainsi que des signatures numériques. Une zone DNSSEC possède au moins une paire de clés (publique/privée) pour signer ses données DNS et fournir ainsi deux services de sécurité essentiels : l'intégrité et l'authenticité des données. Pour faire confiance à des données DNS, un client DNSSEC doit en vérifier les signatures numériques avec la clé de zone appropriée. Cette vérification est basée sur l'établissement d'une chaîne de confiance entre des zones sécurisées. Pour construire cette chaîne, le client a besoin d'un point d'entrée sécurisé : une clé de zone configurée dans le client comme clé de confiance. Puis, le client doit trouver un chemin sécurisé partant de ce point jusqu'aux données DNS demandées. Les clés de zones sont essentielles au fonctionnement de DNSSEC et sont utilisées dans toutes les étapes d'une résolution de nom. Dans ce papier, nous présentons une étude des conséquences d'une clé compromise sur le protocole DNSSEC. Nous décrivons les attaques pouvant être mener grâce à une clé compromise et nous présentons les défenses possibles.

Mots clés : DNSSEC, compromission de clé, securité réseau

1 Introduction

The Domain Name System (DNS) [Moc87a, Moc87b, AL02] is a hierarchical distributed database mostly used to translate host names into IP addresses. The DNS protocol does not include any security services such as data integrity and authentication, which lets some vulnerabilities in the protocol [Bel95, Sch93, AA04]. Therefore, the Internet Engineering Task Force (IETF) has developed the DNS security extensions (DNSSEC).

DNSSEC [Eas99, Gun03, AAL+05a, AAL+05c, AAL+05b] uses public-key cryptography to provide DNS data integrity and authentication. Each node of the DNS tree, called a zone, owns at least a key pair used to generate digital signatures of the DNS zone information. The basic data unit of this information is a resource record (RR). Each RR has a particular type that indicates which kind of data it contains. For example, a DNSKEY RR contains a zone key, a RRSIG RR contains a signature and an A RR contains an IPv4 address.

In order to trust DNS data, a resolver (the DNS client) builds a chain of trust [Gie01] by walking through the DNS tree from a secure entry point [KSL04] (*i.e.*, a trusted key statically configured in the resolver, typically a top level zone) to the queried resource record. A resolver is able to build a chain of trust if it owns a secure entry point and if there are only secure zones on the path from the secure entry point to the queried zone.

In this paper, we describe in Section 2, DNSSEC principles, different types of keys and the signature verification process. Then, in Section 3 we study consequences of a compromised key in DNSSEC. Finally, in Section 4 we present current defenses after a key compromission.

2 DNSSEC architecture and secure name resolution

The Domain Name System (DNS) is a distributed tree-based database. The DNS tree is divided into domains and zones.

2.1 Domains and zones

A domain is a subtree of the DNS tree. The name of a domain is the concatenation of all node's label from the root of the subtree to the root of the DNS tree. Since two nodes, having the same parent node, have a different label, unicity of domain names is ensured. A domain can be include in another domain, for example the `irisa.fr.` domain is include in the `fr.` domain, as shown on Figure 1.

Each domain is constituted of one or several zones. The zone is the administrative unit of the DNS and is represented by a node in the DNS tree. A zone is managed by one or several name servers that store the zone information in a zone file. This zone file contains all the zone resource records.

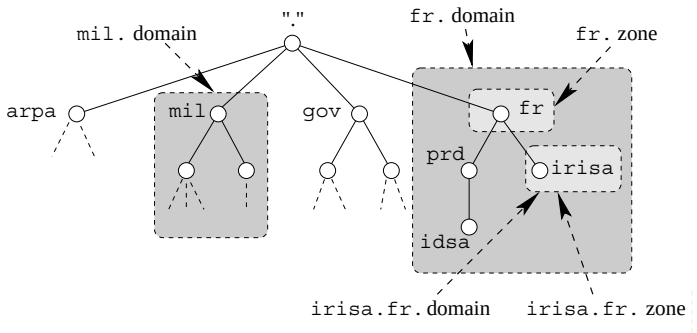


Figure 1: DNS domains and DNS zones.

2.2 DNS entities

Three entities with distinct roles are present in the DNS architecture: the authoritative name server, the cache server and the resolver (see [AL02]).

2.2.1 The authoritative name server

The name server is authoritative on a DNS zone. It stores resource records in its zone file. Every resource record is associated to a DNS name. The name server receives DNS queries about a DNS name and replies with resource records contained in its zone file.

2.2.2 The cache server

A cache server is not authoritative on any zone. It is generally located on a local network to receive queries from local resolvers. A cache server sends responses to these queries using resource records it previously received. If the cache server does not have the answer, it forwards the query to the most appropriated authoritative server it knows. Then, it receives a DNS responses, keeps a copy in memory and sends it to the resolver that has previously launched the name resolution. Using cache servers allows to minimize the burden on authoritative servers [Sit00, JSBM01, CK01].

2.2.3 The resolver

The resolver is the local entity that receives requests from applications and sends DNS queries to name servers or cache servers. After having performed the name resolution, the resolver returns the answer to the application.

2.3 DNSSEC chain of trust

DNS security extensions use public key cryptography and define new resource records to store keys and signatures. Each secure zone owns one or several zone keys. The public part of each key is stored in a DNSKEY resource record. The private part of a zone key is kept secret and should be stored in a secure location. This private part is used to generate a digital signature of each resource record in the zone file. Then, each signature is stored in a RRSIG resource record. To trust a resource record, a resolver must verify at least one signature of this resource record with a trusted zone key.

2.3.1 Two types of DNSSEC keys

There is two way for a resolver to trust a zone key. Either this key is configured in the resolver, this is a *trusted key* (also called a *secure entry point*) or the resolver trusts a DS resource record [Gun03] that authenticates this key. A DS RR is a resource record stored in the parent zone that authenticates a key of the child zone. Basically, a DS RR contains a key identifier and a hash of a child zone public key. The DS RR is signed by parent zone keys.

A DS resource record creates a secure link between a parent zone and its child zone, together with a dependency. Indeed, when a key is renewed in the child zone, the DS RR must be renewed in the parent zone. This implies a communication between the child zone and the parent zone. To minimize the burden due to this additional data exchange, some keys do not have an associated DS RR. The Delegation Signer (DS) model introduces a distinction between two types of keys: the Key Signing Keys (KSK) and the Zone Signing Keys (ZSK). A KSK has an associated DS RR in the parent zone and signs only DNSKEY resource records. A ZSK does not have any associated DS RR and signs all resource records in the zone file.

Even though DS identifies two roles for keys, KSK and ZSK, there is no requirement that zone uses two different keys for these roles. It is expected that many small zones will only use one key, while larger zones will more likely use multiple keys [Gun03].

2.3.2 DNSSEC signature verification process

To decrease the size of the zone file, resource record associated to the same name and having the same type are grouped and signed together. For example, if a zone owns three DNSKEY RRs, these three RRs are grouped in a DNSKEY RRset (or keyset). Then, each key generates a signature for this DNSKEY RRset. Hence, we have three signatures associated to the DNSKEY RRset. The verification of only one signature is sufficient to trust the DNSKEY RRset and hence the three keys it contains.

During a secure name resolution, a resolver builds a chain of trust, that is to say it starts from a trusted key and follows the path to the queried resource records while verifying signatures of resource records and secure link represented by DS-DNSKEY records.

Figure 2 shows different steps a DNSSEC client, having only the root zone key configured as secure entry point (SEP), follows to trust a resource record of the `irisa.fr.` zone.

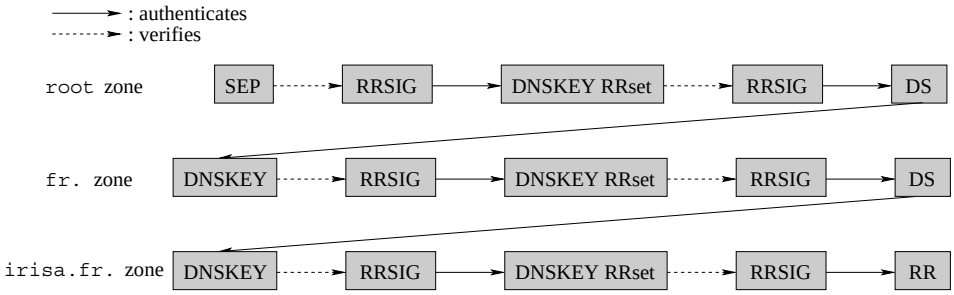


Figure 2: Chain of trust establishment.

At each step, we have the same scheme: a DS resource record allows to trust a KSK, a KSK allows to trust all the zone key (KSK and ZSK) and ZSK allow to trust other resource records of the zone.

After having presented the principles of DNS security extensions, we analyze consequences of a compromised key in DNSSEC and threats induced by the DNSSEC tree-based architecture.

3 Consequences of a compromised key

Key revocation is an expensive service that at least one party will have to pay for. In some circumstances it may be economically preferable to accept the risk of not having a fine-grained revocation service and to require key holding parties to adequately protect their private keys; a route that DNSSEC protocol designers have pursued. There are currently no practical revocation services provided by DNSSEC supporting infrastructures [Cha03].

There are several way to compromise a key. The first one is cryptanalysis. A malicious person gets the private part of a public key only with mathematical knowledges and cryptographical material generated by the key or using vulnerabilities in keys or digital signatures generator. This kind of attack is difficult because cryptographic algorithms and protocols were designed to resist to cryptanalysis. For information, the theoretical time to cryptanalyze a RSA key (usually used in DNSSEC today) is given in table 1 [Odl95].

Basically, private parts of DNSSEC zone keys are stored on a secure location disconnected of any network. If this is not the case, a second way to compromise a key is to access to the private key from the network bypassing all security.

The third possibility is simply when the attacker has a physical access the private keys. The attacker could be, for example, a dishonest or dissatisfied zone administrator.

RSA key size (bits)	MIPS.year
512	3.10^4
768	2.10^8
1024	3.10^{11}
1280	1.10^{14}
1536	3.10^{16}
2048	3.10^{20}

Table 1: RSA key theoretical cryptanalysis time depending on the key size.

Because of the DNS tree-based architecture, when a zone is compromised, all zones in the subdomain of the compromised zone are potentially threatened. An attacker with a compromised zone key can forge resource records cryptographically correct.

We suppose that a zone owns those two types of keys KSK and ZSK. This distinction allows to ease the study without loss of generality. If one key is used as KSK and ZSK at the same time, all properties apply to this key.

3.1 A ZSK is compromised.

When a ZSK is compromised, the attacker that owns the private part of the key can generate digital signatures for any forged resource records. We can notice that signatures generated by the attacker can be verified with the compromised ZSK. This ZSK is itself authenticated by KSKs of the zone because of the existing chain of trust between the zone and its parent zone. Then, the attacker has no need to compromise a KSK of the zone.

3.2 A KSK is compromised.

When a KSK is compromise, the attacker can create and signs DNSKEY RR. Nevertheless, as limiting the scope of a zone key is not a protocol requirement, the attacker can use this KSK to sign other types of resource records. As this KSK has an associated DS RR in its parent zone, this compromised KSK is authenticated by a chain of trust.

3.3 Example of compromised key attack

Assuming the attacker has compromised a key, we suppose the general case where the attacker has no write access to the zone file. As the attacker can not reach the source of the DNS data (the zone file), its target will be to put forged RRs in cache servers. Hence, these false informations will be sent to resolvers by cache servers. This cache pollution becomes as easy as with the DNS protocol, because all the cryptographic material provided by the attacker is correct [LMMM00, GC03].

Indeed, a cache server accepts DNSSEC resource records only if it is capable of building a chain of trust and verifying signatures. However, the chain of trust exists and the attacker can generate signatures of forged records with the compromised key.

DNS messages are identified by a 16 bits identifier in the header of the message. This identifier is set in the query and the associated answer have the same. Resolvers having multiple query waiting at the same time, this identifier allows to associate an answer with a query. So, the only difficulty for the attacker is to force a cache to send a query and then to find the good identifier to answer.

To force the cache server to send a query, the attacker just has to send this query to the cache. Then, the cache behavior is to forward the query to the appropriate authoritative server. Obtaining the good identifier is not a problem because of a birthday paradox with identifiers. It is proved that with only 302 messages the attacker has more than 50% chances to find the good identifier.

With this kind of attack, an attacker can put in a cache server forged records and for example attribute false IP addresses to a web server (a banking account consultation web server...).

In the next Section, we present the way an administrator should follow to defend against compromised key attack.

4 Current defenses against compromised key attacks

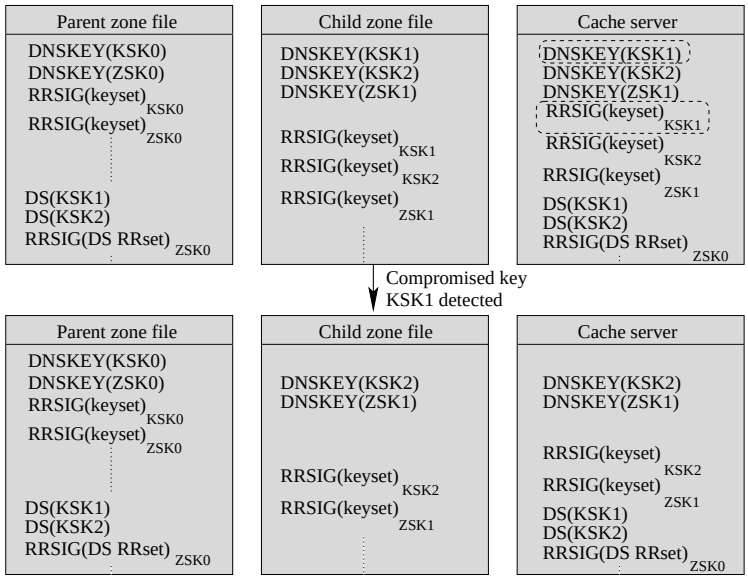
The first thing we can notice to defend against a compromised key is that the administrator must know one of its keys is compromised. This is not obvious because as we have seen, targets of attacks are cache servers and not authoritative servers manage by the administrator. Moreover, the administrator can only act on its own servers, not on the polluted cache.

4.1 Defenses against a compromised KSK

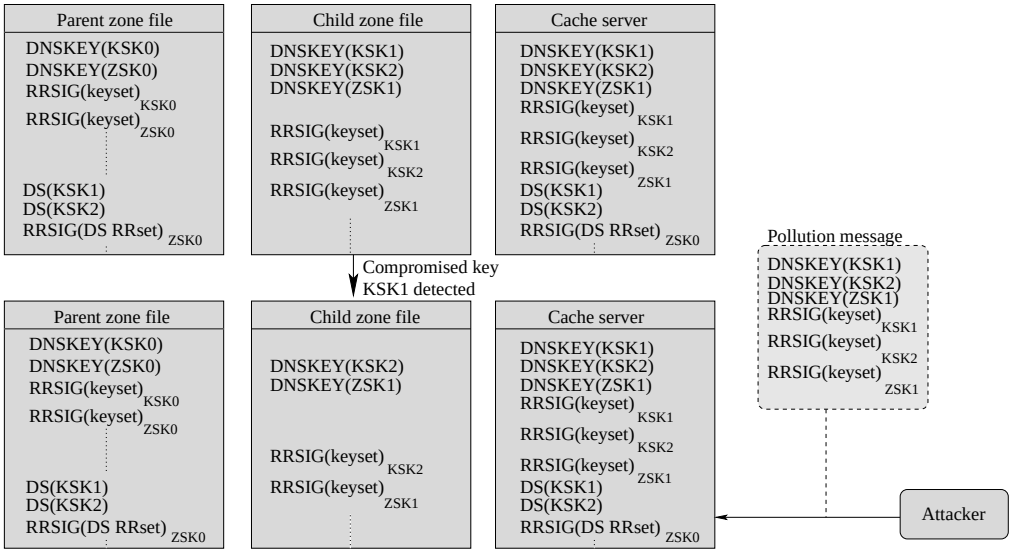
When a zone administrator figures out that one of its key is compromised, he removes immediately this key from its zone file and, as the DNSKEY RRset (group of DNSKEY RRs) has changed, he signs the zone file. Nevertheless, this is not sufficient to stop an attack because of the existing chain of trust (DS-DNSKEY pair) between the parent zone and its child zone as shown on Figure 3. This chain of trust allows to authenticate the compromised key.

Part (a) of the figure 3 shows the state of the child zone file, the state of the parent zone file and the state of the cache server memory. This cache is not polluted. To protect its zone, the child zone administrator has removed the compromised key KSK1 and has signed its zone file. A resolver sending a query to this cache server will obtain a DNSKEY RRset with KSK2 and ZSK1 (keys deployed in the zone).

Part (b) of the figure 3 shows the state of the cache server when the pollution attack succeeds. We notice that even if the compromised key is removed from the zone file, it remains in the cache server. Hence, even if parent and child zone files are in a consistent



(a) Without cache pollution



(b) With cache pollution

Figure 3: States of child zone file, parent zone file and cache server.

state, a resolver sending a query to the cache server will receive the compromised key KSK1 and signatures generated by KSK1. Then, this resolver can build a chain of trust, including DS(KSK1) and KSK1, authenticating the compromised key and forged RR signed by this key. Remove the compromised key is not sufficient because of the DS record authenticating the compromised key ant then the existence of a chain of trust.

To remove this chain of trust, the associated DS RR must be removed too as soon as possible. Nevertheless, Figure 4 shows this does not stop the attack immediately.

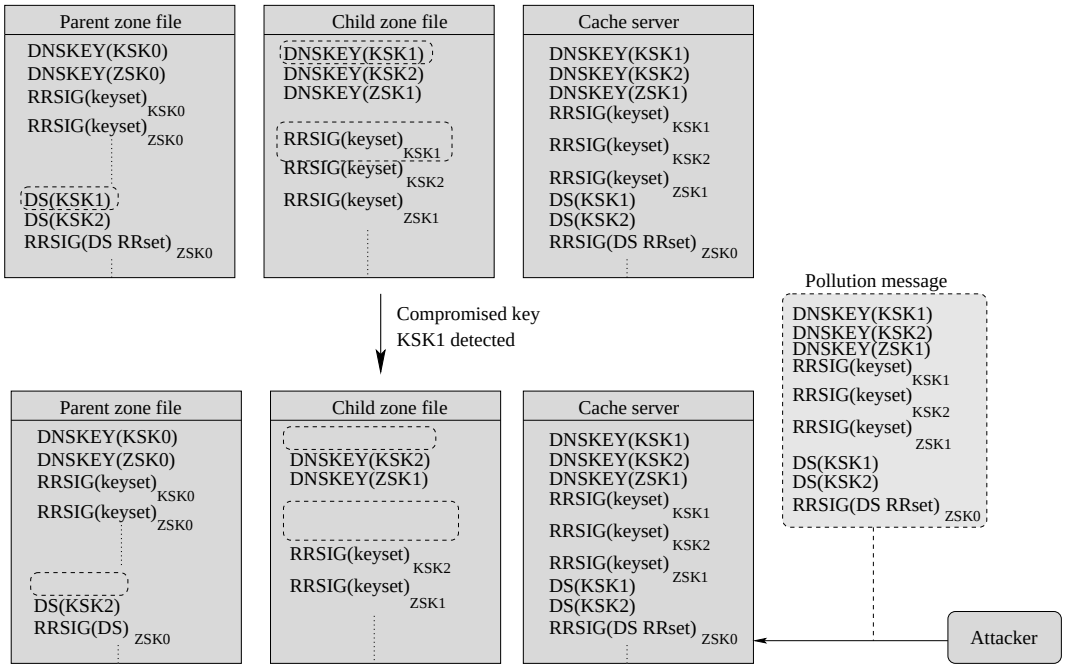


Figure 4: Cache pollution when compromised keys and DS are removed from zone files.

When the attacker pollutes a cache server, he adds this DS RR to the pollution message to bypass the DS RR removing. Consequently, a resolver sending a query to this polluted cache server can build a chain of trust authenticating the compromised key. This is due to that something true in DNSSEC (a RR and its signature) at a time is true until the signature expiration and to the multiple way to build a chain of trust (each DS-DNSKEY pair). Hence, a cache pollution attack remains valid until:

$$\max_{k \in K, p \in P} (expiration(RRSIG(DNSKEY\ \ RRset)_k), expiration(RRSIG(DS_c)_p),$$

where K is the set of KSKs of the zone that are not compromised, P is the set of ZSKs of the parent zone, DS_c the DS RR authenticating to the compromised key and *expiration* is the expiration date of the signature. This formula means: the expiration of signatures of all (DS-DNSKEY) pair representing a chain of trust. After this delay, the only signature the attacker can create is generated with the compromised key. As all chain of trust have been removed, a resolver cannot authenticate the compromised key. The compromised key has become useless.

4.2 Defenses against a compromised ZSK

When a ZSK is compromised, even if the compromised key is removed from the zone file, a resolver using the polluted cache server (*i.e.*, containing the old DNSKEY RRset with the compromised key) could use the compromised key until signatures generated by KSKs expire. We can notice that there is no need to remove KSK or DS from the zone file. Once the compromised ZSK is removed, the DNSKEY RRset is modified and hence signatures too (see paragraph 2.3.1). Consequently, the attack will succeed until all old DNSKEY signatures expiration, that is to say:

$$\max_{k \in K} (expiration(RRSIG(DNSKEY\ \ RRset)_k))$$

where K is the set of KSK and *expiration* is the expiration date of the signature.

This formula means the expiration of all signatures generated by a KSK and hence means the deletion of all chain of trust authenticating the compromised key. As the attacker cannot generate a new signature of the DNSKEY RRset with a key having an associated DS RR (a KSK), after the expiration of signatures, the compromised key becomes useless.

The validity period of a signature [KG06] is generally one week or one month depending on the key type and the key lifespan is one month for a ZSK and one year for a KSK. It appears that the time an attacker has to launch attacks with a compromised key is not negligible.

5 Conclusion

The DNS security extensions (DNSSEC) are based on public-key cryptography. In this paper, we focused on consequences of a compromised key in DNSSEC. It appears that this is a problem with major impact, because of the tree architecture of the DNS. When a zone key is compromised, all zones having the compromised zone as ancestor are threatened. The attacker that owns a compromised key of a zone can forge resource records for any of its subzone and then pollute recursive cache servers with these records. Moreover, even if the compromised key is immediately removed from its zone file, the compromised RR may remain in cache servers for a long time. In this paper, we have given formulas to find the period during which a cache pollution attack with a compromised key can succeed.

References

[AA04] D. Atkins and R. Austein. Threat Analysis of the Domain Name System. RFC 3833, August 2004.

[AAL+05a] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS Security Introduction and Requirements. RFC 4033, March 2005.

[AAL+05b] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. Protocol Modifications for the DNS Security Extensions. RFC 4035, March 2005.

[AAL+05c] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. Resource Records for the DNS Security Extensions. RFC 4034, March 2005.

[AL02] P. Albitz and C. Liu. *DNS and BIND.* O'Reilly & Associates, Inc., Sebastopol, Californie, 4th edition, January 2002.

[Bel95] S. M. Bellovin. Using the Domain Name System for System Break-Ins. In *Proceedings of the 5th Usenix UNIX Security Symposium*, pages 199–208, June 1995.

[Cha03] B. Chan. Identity-Based PKI for DNSSEC. Master's Thesis, Royal Holloway University of London, 2003.

[CK01] E. Cohen and H. Kaplan. Proactive Caching of DNS Records: Addressing a Performance Bottleneck. In *Symposium on Applications and the Internet*, pages 85–94, January 2001.

[Eas99] D. Eastlake. Domain Name System Security Extensions. RFC 2535, March 1999.

[GC03] G. Guette and B. Cousin. Les faiblesses du DNS. In *2ème rencontre francophone sur la Sécurité et Architecture Réseaux (SAR)*, pages 235–244, July 2003.

[Gie01] R. Gieben. Chain of trust: The parent-child and keyholder-keysigner relations and their communication in dnssec. Master's thesis, University of Nijmegen, 2001.

[Gun03] O. Gundmundsson. Delegation Signer Resource Record. RFC 3658, December 2003.

[JSBM01] J. Jung, E. Sit, H. Balakrishan, and R. Morris. DNS Performance and the Effectiveness of Caching. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop '01*, pages 153–167, November 2001.

[KG06] O. Kolkman and R. Gieben. DNSSEC operational practices. Draft IETF, work in progress, February 2006.

[KSL04] O. Kolkman, J. Schlyter, and E. Lewis. Domain Name System KEY (DNSKEY) Resource Record (RR) Secure Entry Point (SEP) Flag. RFC 3757, April 2004.

[LMMM00] A. Lioy, F. Maino, M. Marian, and D. Mazzocchi. DNS Security. In *Terena Networking Conference*, May 2000.

[Moc87a] P. Mockapetris. Domain Names - Concept and Facilities. RFC 1034, November 1987.

[Moc87b] P. Mockapetris. Domain Names - Implementation and Specification. RFC 1035, November 1987.

[Odl95] A. M. Odlyzko. The future of integer factorization. *CryptoBytes (The technical newsletter of RSA Laboratories)*, 1(2):5–12, 1995.

[Sch93] C. L. Schuba. Addressing Weaknesses in the Domain Name System. Master's thesis, Purdue University, Department of Computer Sciences, August 1993.

[Sit00] E. Sit. A Study of Caching in the Internet Domain Name System. Master's Thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Sciences, May 2000.