INTERNET-DRAFT R. Gieben
DNSEXT Working Group NLnet Labs
Expires September 2001 T. Lindgreen
NLnet Labs

Parent's SIG over child's KEY

draft-ietf-dnsext-parent-sig-00.txt

Status of This Document

Comments should be sent to the authors or the DNSEXT WG mailing list namedroppers@ops.ietf.org.

Copyright Notice

Abstract

When dealing with large amounts of keys the procedures to update a zone and to sign a zone need to be clearly defined and practically possible. The current idea is to have the KEY RR and the parent's SIG to reside in the child's zone and perhaps also in the parent's zone. We feel that this would lead to very complicated procedures for large TLDs. We propose an alternative scheme in which the parent zone stores the parent's signature over the child's key and also a copy of the child's key itself.

The advantage of this proposal is that all signatures signed by a key are in the same zone file as the producing key. This allows for a simple key rollover and resigning mechanism. For large TLDs this is extremely important.

We further discuss the impact on a secure aware resolver/forwarder and the impact on the authority of KEYs and the NXT record.

Table of Contents

1. Introduction

Within a CENTR working group NLnet Labs is researching the impact of DNSSEC on the ccTLDs and gTLDs.

In this document we are considering a secure zone, somewhere under a secure entry point and on-tree [1] validation between the secure entry point and the zone in question. The resolver we are considering is security aware and is preconfigured with the KEY of the secure entry point.

RFC 2535 [3] states that a zone key must be present in the apex of a zone. This can be in the at the delegation point in the parent's zonefile (normally the case for null keys), or in the child's zonefile, or in both. This key is only valid if it is signed by the parent, so there is also the question where this signature is located.

The original idea was to have the KEY RR and the parent's SIG to reside in the child's zone and perhaps also in the parent's zone. There is a draft proposal [4], that describes how a keyrollover can be handled.

At NLnet Labs we found that storing the parent's signature over the child's key in the child's zone:

- makes resigning a KEY by the parent difficult

- makes a scheduled keyrollover very complicated
- makes an unscheduled keyrollover virtually impossible

We propose an alternative scheme in which the parent's signature over the child's key is only stored in the parent's zone, i.e. where the signing key resides. This would solve the above problems.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [2].

2. Proposal

The core of the new proposal is that the parent zone stores the parent's signature over the child's key and also a copy of the child's key itself. The child zone also contains its zonekey, where it is selfsigned.

The advantage of this proposal is that all signatures signed by a key are in the same zone file as the producing key. This allows for a simple key rollover and resigning mechanism. For large TLD's this is extremely important. A disadvantage would be that not all the information concerning one zone is stored at that zone, namely the (parent) SIG RR. Note that the same argument can be applied to a zone's NULL key, which is also stored at the parent.

3. Impact on a secure aware resolver/forwarder

The resolver must be aware of the fact that the parent is more authoritative than a child when it comes to deciding whether a zone is secured or not.

Without caching and with on-tree validation, a resolver will always start its search at a secure entry point. In this way it can determine whether it must expect SIG records or not.

Considering caching in a secure aware resolver or forwarder. If information of a secure zone is cached, its validated KEY should also be cached.

If the KEY record expires, because the KEY TTL expires or because the SIG is no longer valid, the KEY should be discarded. The resolver or forwarder should then also discard other data concerning the zone because it is no longer validated and possible bad data should not be cached.

3.1. Impact of key rollovers on resolver/forwarder

When a zone is in the process of a key rollover, there could be a discrepancy between the KEY and the SIG in the apex of the zone and the KEY and SIG that are stored in the cache of a resolver.

Suppose a resolver has cached the NS, KEY and SIG records of a zone. Next a request comes for an A record in that zone. Also the zone is in the process of a keyrollover and already has new keys in its zone. The resolver receives an answer consisting of the A record and a SIG over the A record. It uses the tag field in the SIG to determine if it has a KEY which is suitable to validate the SIG. If it does not has such a KEY the resolver must ask the parent of the zone for a new KEY and then try it again. Now the resolver has 2 keys for the zone, according to the tag field in the SIG it can use either one.

If the new key also does not validate the SIG the zone is marked bad. If the KEY found at the parent is the NULL key the resolver knows that the child is considered insecure. This could for instance be in the case the private key of the zone is stolen.

4. Key rollovers

Private keys can be stolen or a key can become over used. In both cases a new key must be signed and distributed. This event is called keyrollover. We further distinguish between a scheduled and an unscheduled key rollover. A scheduled rollover is announced before hand. An unscheduled key rollover is needed when a private key is compromised.

4.1. Scheduled key rollover

When the signatures, produced by the key to be rolled over, are all in one zone file, there are two parties involved. Let us look at an example where a TLD rolls over its zone key. The new key needs to be signed with the root's key before it can be used to sign the TLD zone and the zone keys of the TLD's children. The steps that need to be taken by TLD and root are:

- the TLD adds the new key to its keyset in its zonefile. This zone and keyset are signed with the old zonekey
- then the TLD signals the parent
- the root copies the new keyset, consisting of the both new and the old key, in its zonefile, resigns it and signals the TLD
- the TLD removes the old key from its keyset, resigns its zone with the new key, and signals the the root
- the root copies the new keyset, now consisting of the new key only, and resigns it

4.2. Unscheduled key rollover

Although nobody hopes that this will ever happen, we must be able to cope with possible key compromises. When such an event occurs, an immediate keyrollover is needed and must be completed in the shortest possible time. With two parties involved, it will still be awkward, but not impossible to update two zonefiles overnight. "Out-of-band" communication between the two parties will be necessary, since the compromised old key can not be trusted. We think that between two

parties this is doable, but this complicated procedure is beyond the scope of this document. [5]

5. Zone resigning

Resigning a TLD is necessary before the current signatures expire. When all SIG records, produced by the TLD's zone key are kept in the TLD's zonefile, and only there, such a resign session is trivial, as only one party (the TLD) and one zonefile is involved.

6. Consequences for KEY and NXT records

A key record is only present in a child zone to facilitate a key rollover. A resolver should therefore be aware that the zonekey of a child zone is actually stored in the parent's zone. This also affects the NXT record and the authority of KEY resource records.

6.1. KEY bit in NXT records

RFC 2535 [3], section 5.2 states:

" The NXT RR type bit map format currently defined is one bit per RR type present for the owner name. A one bit indicates that at least one RR of that type is present for the owner name. A zero indicates that no such RR is present. [....] "

With a KEY still present in a child zone we do not see a compelling reason to change this default behavior.

6.2. Authority of KEY records

The parent of a zone generates the signature for the key belonging to that zone. By making that signature available the parent publicly states that the child zone is trustworthy: when it comes to security in DNSSEC the parent is more authoritative than the child.

From this we conclude that a parent zone MUST set the authority bit to 1 and child zones MUST set this bit to 0 when dealing with KEYs from that child zone.

A secure entry point has a selfsigned key and thus has no parent who is more authoritative on that key. This is not a problem. If a resolver knows that a secure entry point is a secure entry point it must have its key preconfigured. There is no need for a parent in this scenario, because the resolver itself can check the security of that zone. A interesting consequence of this is that nobody, but the resolver is authoritative for a key belonging to a secure entry point. This authority must established via some out of band mechanism, like publishing keys in a newspaper.

7. Security Considerations

This whole document is about security.

Authors' Addresses

R. Gieben
Stichting NLnet Labs
Kruislaan 419
1098 VA Amsterdam
miek@nlnetlabs.nl

T. Lindgreen
Stichting NLnet Labs
Kruislaan 419
1098 VA Amsterdam
ted@nlnetlabs.nl

References

[1] Lewis, E. "DNS Security Extension Clarification on Zone Status",
www.ietf.org/internet-drafts/draft-ietf-dnsext-zone-status-04.txt
[2] Bradner, S. "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119
www.ietf.org/rfc/rfc2119.txt
[3] Eastlake, D. "DNS Security Extensions", RFC 2535
www.ietf.org/rfc/rfc2535.txt
[4] Andrews, M., Eastlake, D. "Domain Name System (DNS) Security Key Rollover"
www.ietf.org/internet-drafts/draft-ietf-dnsop-rollover-01.txt
[5] Gieben, R. "Chain of trust"
secnl.nlnetlabs.nl/thesis/thesis.html

Full Copyright Statement