INTERNET-DRAFT
DNSEXT Working Group
Expires September 2001

R. Gieben
NLnet Labs
T. Lindgreen
NLnet Labs

Parent stores the child's zone KEYs

draft-ietf-dnsext-parent-stores-zone-keys-01.txt

Status of This Document

This document updates RFC 2535.

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 zone KEY RR and the parent's SIG to reside in the child's zone and perhaps also in the parent's zone. Operational experiences have prompted us to develop an alternative scheme in which the parent zone stores the parent's signature over the child's key and also the child's key itself.

The advantage of this scheme 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.

Besides the operational advantages, this also obsoletes the NULL key, as the absence of child's zone KEY, which is securely verified by the absence of the KEY-bit in the corresponding NXT RR, now unambiguously indicates that the child is not secured by this parent.

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 [RFC 3090] 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. We also make a distinction between a scheduled and a unscheduled key rollover. A scheduled rollover is announced before hand. An unscheduled key rollover is needed when a private key is compromised.

RFC 2535 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, 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 and this zone KEY are located.

The original idea was to have the zone 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 [RFC 2535], that describes how a keyrollover can be handled.

At NLnet Labs we found that storing the parent's signature over the child's zone 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 zone KEY and the child's zone KEY itself are only stored in the parent's zone, i.e. where the signing key resides. This would solve the above problems and also obsoletes the NULL KEY.

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 RFC 2119.

2. Proposal

The core of the new proposal is that the parent zone stores the parent's signature over the child's zone KEY and also the child's zone KEY itself, and is authoritative for both KEY and SIG. The child zone may also contain its zone KEY, in which case is must be selfsigned. The child zone must not hold the parent's SIG, and must also not set the AA-bit on requests for its zone KEY.

The main 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. A disadvantage would be that not all the information concerning one zone is stored at that zone, this is

covered in section 7.2.

A parent running DNSSEC SHOULD NOT refuse a request from a child to include and sign its key, but can ask for certain conditions to be met. These condition could include a fee, sufficient authentication, signing a non liability clause, the conditions specified in section 8 of this document, etc.

2.1. TTL of the KEY and SIG at the parent

Each zone in DNS expresses in its SOA record the maximum and minimum

TTL values that they allow in the zone. Thus it is possible that the parent will sign with a value that is unacceptable to the child. The parent MUST follow the TTL request of the child as long as that is within the allowed range for the parent.

2.2. No NULL KEY

This proposal obsoletes the NULL KEY. If there is no child KEY at the parent, which can be securely verified by inspecting the the unset KEY-bit in the corresponding NXT RR, the child is not secured by this parent (of course, the child can then still be secured off-tree). This updates section 3.1.2 "The zone KEY RR Flag Field" of RFC 2535, it says:

" 11: If both bits are one, the "no key" value, there is no key
information and the RR stops after the algorithm octet.
By the use of this "no key" value, a signed zone KEY RR can
authenticatably assert that, for example, a zone is not
secured. See section 3.4 below. "

As we don't have a NULL KEY anymore this is obsoleted. Section 3.4 "Determination of Zone Secure/Unsecured Status":

" A zone KEY RR with the "no-key" type field value (both key type flag bits 0 and 1 on) indicates that the zone named is unsecured while a zone KEY RR with a key present indicates that the zone named is secure. The secured versus unsecured status of a zone may vary with different cryptographic algorithms. Even for the same algorithm, conflicting zone KEY RRs may be present. "

This is rewritten as:

" A zone is considered secured by on-tree validation [RFC 3090] when the there is a zone KEY from that zone present at its parent. If there is no zone KEY present, and the resolver is also unaware of alternative algorithms used and/or possible off-tree validation, the

zone is considered unsecured. "

To further clarify this. A zone is secure, when the resolver expects it to be, there are two possibilities:

1. When its parent is secure and holds a signed KEY for this child.
2. When zone is a secure entry point, i.e. the resolver is preconfigured with the KEY of this zone.

RFC 3090 calls this globally secured.

When a zone contains SIGs and a selfsigned KEY and this KEY is preconfigured in the resolvers of interest, the a zone can be considered locally secured (the RFC 3090 defintion). hijacked.

If a zone is not globally or locally it must be considered unsecure.

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.

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 key rollover 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 parent indicates by having a not set KEY-bit in the NXT RR that there is no KEY for this zone, the child must be considered unsecured by this parent, despite the appearance of an (old) KEY in the cache. This could for instance happen after an emergency request from the child, who has suffered a key compromise, and has decided to prefer being unsecured over either dropping of the Internet or being exposed to have verifiable secure info added by the key-compromiser to their zone information.

4. 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 possible 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 KEY set in its zonefile. This zone and KEY set are signed with the old zone KEY
- then the TLD signals the parent

- the root copies the new KEY set, 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 KEY set, resigns its zone with the new KEY, and signals the the root
- the root copies the new KEY set, now consisting of the new key only, and resigns it

Note that this procedure is immune to fake signals and spoofing attacks (as long as there is no key compromise):

- on a fake signal either way the action becomes a null-action as the new KEY set is identical to the existing one.
- a spoofed new KEY set will not validate with the existing KEY that the parent holds.

5. 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.

An alternative to an emergency key-rollover is becoming unsecured as an emergency measure. This has already been mentioned above in section 3.1. This only involves an emergency change in the parents zonefile (deleting the child's zone KEY), and allows the child and its underlying zones time to clean up before becoming secured again, without dropping from the Internet or being exposed to having secured but false zone information.

6. Zone resigning

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

7. Consequences for KEY and NXT records

There are two reasons to have of the child's zone KEY not only at the parent but also in the child's own zonefile:

1. to facilitate a key-rollover
2. to prevent local lookups for local information to suffer from possible loss of access to its outside parent

To cope with 1, secure aware resolvers MUST be aware that during a key-rollover there may be a conflict, and that in that case the

parent always holds the active KEY set. To cope with 2, the local resolver/caching forwarder should be preconfigured with the zone-KEY and thus looks at its own zone as were it a secure entry-point. For both things to work, the zone-KEY set must be selfsigned in the child zonefile.

7.1. KEY bit in NXT records

RFC 2535, 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. [....] "

As the zone KEY is present in a child zone, and signed by the zone KEY (thus selfsigned), the definition of NXT RR type bit states in RFC 2535, section 5.2 that the KEY bit must be set. We do not see a

compelling reason to change this default behavior.

7.2. 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.This also causes resolvers to pick up and cache the right KEY set, in case it finds conflicting KEY sets during a key-rollover.

Some zones have no parent to make it authoritatively secure, for instance, the root. To be secure anyway it must be defined a secure entry point. 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 is authoritative for a key belonging to a secure entry point. This authority must established via some out of band mechanism, like publishing it in a newspaper.

7.3. Selecting KEY sets

As the zone KEY set is present in two places, there is a possibility of two conflicting KEY sets, this will happen during a key-rollover and may happen at other times.

With one exception, a resolver MUST always select the KEY set from the parent in case of a conflict, as this is the active KEY set. For this reason, the parent sets the AA-bit on requests, while the child does not.

The one exception is when a resolver regards the child's zone as a secure-entry point, in which case it has the zone KEY preconfigured. In other words: a preconfigured KEY has even more authority then what a parent says. Specifying a zone as a secure entry-point makes sense for a local resolver in its own local zone.

8. The zone KEY and local KEY records.

It must be recognized that the zone KEY RR, which is signed by a non-local organization, is something special. The external signature over the public part of the key provides the local zone-administrator

with the authority to use the corresponding private part to sign everything local, and thus to make his/her own zone secure. Please also note that the external signer, and NOT the local zone is authoritative for the zone KEY RRset.

Part of the RRs that the zone-administrator may wish to sign are KEY RRs for local use, for instance for IPSEC.

To make sure, that the local zone is authoritative for its own local KEY RRs, and that they get not exported and signed externally, these local KEY records SHOULD not be part of the zone KEY RRset. Therefore, they could be placed under a label in the zonefile, f.i. keys.child.parent, or for these kind of keys a new RR type could be defined (e.g. PUBKEY).

Besides being kept clear of local KEY records, the zone KEY RRset SHOULD also be kept clear of any other obsolete or otherwise not strictly needed KEY records, because this increases the number of possible key compromise attacks and also increases the size of the parents zone file unnecessarily.

In other words: the KEY RRset with the toplevel label of a zone SHOULD only contain its active zone KEY, unless a key-rollover is in progress. During a keyrollover a new KEY RR must be added to this RRset. Once the new KEY becomes the active zone KEY, the old KEY becomes obsolete and SHOULD be removed as soon as practically possible. Information stored in caches SHOULD NOT be an issue on when to remove the old zone KEY.

9. Security Considerations

This document addresses the operational difficulties that arise when DNSSEC is deployed. By putting the child's zone KEY at the parent we solve at lot of problems by minimizing the amount of communication between the two. There is one security issue: the parent must not ever create a valid parental SIG over a KEY RR, from which the private part is (also) known to someone else than the legitimate administrator of the child zone. This can happen in two ways:

1. The private KEY at the child has been compromised.
2. The parent has been fooled and thus insufficiently checked

whether the KEY RR is really from the child.

For the security it doesn't matter if the SIG and the KEY are located at the child or at the parent, but if they are located at the parent it is much easier to replace the SIG. And by keeping the parental SIG

lifetime short, the parent helps to protect the child against possible key compromises. The selfsigned zone KEY stored in the child's zone can have a long SIG expiration lifetime, this has no impact on the child's security.

All security considerations from RFC 2535 apply.

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

[RFC 3090] Lewis, E. "DNS Security Extension Clarification on Zone Status", RFC 3090
www.ietf.org/rfc/rfc3090.txt

[RFC 2119] Bradner, S. "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119
www.ietf.org/rfc/rfc2119.txt

[RFC 2535] Eastlake, D. "DNS Security Extensions", RFC 2535
www.ietf.org/rfc/rfc2535.txt

Full Copyright Statement