Resolver Analysis for a Signed Root

NLnet Labs document 2009-002 version 06. June 22, 2009

Abstract

The traffic effects for resolvers are calculated in case the DNS root zone is signed. The caching resolver traffic with its stub resolvers changes very little. The caching resolver traffic with the root authority servers increases by about a factor five. For the resolver operator the actual absolute traffic increase is very limited because most of the resolver traffic does not go to the root servers and therefore does not change. For the root operator the traffic increase is substantial.

Introduction

This document examines the effects on a resolver if the DNS root zone is signed with DNSSEC. The DNS (Domain Name System) is used to translate names into numbers, for example translating www.google.com into a computer address where a web browser can fetch data. DNSSEC is a system for using public key cryptography to provide digital signatures over data in the DNS. These signatures and other data for DNSSEC take up space, therefore traffic increases in size.

The DNS root zone is the starting point for a resolver that attempts to look up a name. The root zone is served by the root zone authoritative DNS servers. These servers publish the DNS information for the root zone. For each of the toplevel domains the root zone authoritative DNS servers give delegations to the responsible DNS servers that can give more information about those toplevel domains. *Toplevel* domains are the last word in a domain name: .com, .org, .uk, .nl, ... for a total of 280 toplevel domains in the current root zone.

The authoritative DNS servers that publish the root zone data do not contain the answers to all the possible questions. The root servers refer the resolver to another authoritative DNS server, that knows more about the question. For that reason the authoritative DNS servers for the root have a list of servers to refer to for every toplevel domain name. Providing this data in an answer is called giving a *referral*. The data itself is called a *delegation*.

Sometimes queries are made for toplevel domains that do not exist. In these cases the authoritative DNS servers for the root respond with a negative reply to the resolver. This reply indicates the nonexistence of the query. This is called a *nxdomain* reply, referring to the error code used in the message (NX domain: Non-eXistent domain).

A *resolver* is a DNS server that performs lookups towards the DNS servers on the internet. First to the root servers and then delegated onwards towards an authoritative DNS server on the internet with the answer. The resolver has a cache to remember data, and to speed up lookups. This is important because it means that queries that can be answered from the cache do not need traffic from the resolver to the authoritative DNS servers for the root zone. The cache behaviour of a resolver is simulated to calculate how many of the queries necessitate traffic to the servers for the root zone.

Methodology

DNSSEC is backwards compatible. This means that older resolvers see no difference, they get the exact same reply as before. DNSSEC-capable software sets the DO (DNSSEC OK) flag inside queries, and the DNSSEC information is returned to them. If the resolver does not set the DO flag nothing changes for it and the computers that send queries to it. This document examines resolvers that turn the DO flag on in the queries they send.

The change in size of the answers from the root servers is different for referral answers and nxdomain answers, because DNSSEC uses different data to provide digital signatures for them. Thus to estimate the traffic size change, the size increase for the different types of replies by the root servers must be estimated. Also, the relative frequency of these different types of messages has to be measured.

The size increase for different types of replies is calculated below. This calculation is based on the test signed root available. Using a query trace from a resolver at a large ISP, the relative frequency of the different messages

is estimated in the section after that.
Below, the size increase of answers is examined. This is followed by a simulation of the cache. After that the traffic increase is calculated using the results from the earlier sections. Finally, the impact of the size increase on cache efficiency is discussed and the results are summarised.

Larger Answers From the Root

The answers given by the root servers are bigger if the resolver sets the DO flag. The average is taken over the 280 toplevel domains in the root. The results are calculated by performing a scripted lookup for every one of the toplevel domain names, using the 'dig' tool from the BIND utilities suite, to both k.root-servers.net and to the IANA test root, and comparing the size of the result. The results are then averaged.
The average size of a reply with a delegation for the unsigned root zone is 297 bytes. With a 1024 bit RSA key for signatures the average delegation reply becomes 489 bytes, 192 bytes larger. It was measured using the query names 'example.tld' for the 280 toplevel domains in the root zone. This is a size increase of 1.65x.
The average size of an answer for nonexistent names for the unsigned root zone is 122 bytes. With a 1024 bit RSA key for signatures the average answer for nonexistent names becomes 625 bytes, 502 bytes larger. This was tested for query names 'tldexample', for the 280 toplevel domains in the root zone. This exercises all of the combinations of DNSSEC records given in reply to nxdomain queries to the root. This is a size increase of 5.12x.
Apart from delegations and DNSSEC data for nxdomain answers, the root zone also contains a small amount of data that describes the DNS root zone itself. When a resolver starts it needs to perform a root priming query to obtain this information. The answer to the root priming query is larger as well, however, this query is only done once. Also DNSKEY queries to the root are performed once, if the resolver performs DNSSEC validation. The traffic of these two queries is assumed to result in only a couple of Kbs extra per day for the resolver.
If the root is signed with a larger zone signing key (ZSK) the responses measured in this section increase in size.

The Resolver Cache Simulation

To see which queries are sent to the root, in this section a resolver query trace is examined. The frequency of particular queries to the root is noted. Also the behaviour of the resolver cache is simulated.
The trace contains the (anonymized) queries sent by end users to the resolver at the ISP. For example, in the trace websites such as 'www.google.com' appear. Also the query type is given, such as 'A' to retrieve an IPv4 address. Popular websites appear more often than unpopular ones.
When the query is the root priming query or asks other data on the root itself, it is labelled *root data*. When the last word of the query name is an existing toplevel domain from the root zone, this is labelled a *referral*. Other queries, where the last word of the query name does not exist in the root zone, are labelled queries for *nonexistent* toplevel domains.
The resolver caches responses and thus not every query ends up becoming a query sent to the root server. For root data, the result is cached and the query (for the specific record type) only has to be done once and later queries can be answered from the cache. For referrals, once the delegation data is stored in the cache, further queries with the same toplevel domain do not need to contact the root servers again. For queries for nonexistent toplevels the root servers have to be contacted to learn the answer for every specific query name and query type.
The simulation was written in roughly 200 lines of python and keeps track of the root information in the resolver cache. Every query is classified by type. Then it is seen if the query is answered from cache. If not, a new cache item is created using root information from a file. The number of queries is printed by type, and the number of cache misses is printed. The cache misses equal the number of queries towards root servers because the simulation only caches root information.
For the simulation the TTLs (Time To Live for the resource records) from the root are assumed to be long enough so that queries that are cached are not asked for again. Also the size of the cache memory is assumed large enough to hold all the information that the resolver needs to cache. These are reasonable assumptions, the current TTLs used on the root data are 48 hours and longer. The total amount of data in the root zone is reasonably small, about 100 kilobytes, which should fit in the cache memory of a resolver.
The queries that are listed in the trace come from *stub resolvers*. These are the programs used by web-browsers

to contact the resolver, to ask the address for a name. Thus there is data traffic between the stub resolvers and the resolver under examination (called a *caching resolver* in this text, even though some stub resolvers cache too). For some queries the caching resolver needs to contact the root servers to obtain data.

Table 1. *Queries at a caching resolver classified for root server impact. Incoming and outgoing queries are detailed into root data, referral and nonexistent toplevel domain answers.*

	From stub	*Stub queries that become a root query*	*Queries sent to the root*
Referrals	95.95%	0.002%	0.51%
Root data	0.03%	0.000%	0.01%
Nonexistent	4.02%	0.468%	99.48%
Total queries	100.00%	0.47%	100.00%

The first column of Table 1 shows the number of queries asked by stub resolvers to the caching resolver. The second column shows the number of those stub queries that resulted in a query to the root servers. Because some of these numbers are exceedingly small, the last column shows the relative makeup of the queries sent to the root.

The first row shows queries that are for existing top level domains, and thus the root servers reply with a referral to the DNS servers for those top level domains. These are cached very well, 0.51% of the queries sent to the root are used to answer 95.95% of the queries from stubs.

The second row shows queries for data directly present in the root. This number includes root priming queries sent by stub resolvers. Stub resolvers do not need to perform root priming, perhaps these are caused by administrators debugging their network or people running checks on the proper functioning of the root system. The amount is small compared to the rest of the traffic. Therefore this amount is disregarded in the traffic calculations below.

The third row shows queries for nonexistent names. The toplevel domain in the name does not exist, and thus the resolver has to ask the root servers for every query name to see if it exists or not. Although this makes up only 4.02% of the queries from stubs, they result in 99.48% of the queries sent to the root server. This is because they are cached per query name and query type whereas the referrals can be cached per toplevel domain name.

By comparing the number of queries that become a root query versus the number of incoming queries the cache hit rate can be calculated. The resolver cache stops 99.997% of referrals from going to the root servers. The cache stops 99.81% of direct data queries from going to root servers. The cache stops 88.34% of the queries which result in an nx-domain return code because the toplevel domain does not exist.

The stub resolvers gave EDNS records in some queries, the trace that was available did not say if the DO flag was set but only notes EDNS presence, it is assumed here that all the EDNS enabled queries have the DO flag turned on. In reality not all would have the DO flag set, and this would make the following percentages even smaller. Stub resolvers included an EDNS record in 1.6% of all queries, but included an EDNS record in 0.19% of the queries for nonexistent toplevel domains. It is assumed that the number of DO flag queries has remained similar, however it is likely that due to software updates more software sets the DO flag now.

Increased Traffic

In the previous two sections the size increase per message and the frequencies of particular messages have been calculated. By putting them together, the traffic increases for the caching resolver can be estimated. The calculation is performed per traffic type. Queries for data directly about the root itself, such as root priming queries, are assumed to be so infrequent they can be ignored.

The query stream towards the root is increased by the referrals and queries for nonexistent toplevel domains. Thus, 0.51% of the traffic is referrals, and these increase by 1.65x, and 99.48% of the traffic towards the root is for nonexistent names, and these increase by 5.12x. The average increase for traffic from the resolver towards

the root is therefore 5.10x. If the root servers mostly receive traffic from resolvers at ISPs, then for the root servers traffic goes up by about 5x.
For the query stream from the stub resolvers, the fraction of stub resolvers that set the DO flag has also to be taken into account. From the 95.95% of queries that are answered from the root by a referral, the caching resolver replies with data obtained from other authority servers, which are assumed to remain the same for this analysis. If those other authority servers deploy DNSSEC, those answers would increase in size as well. The nonexistent toplevel domain answers are repeated verbatim by the caching resolver, if the DO flag is set by the stub resolver. For 4.02% of queries, 0.19% has a DO flag, and they increase 5.12x. This results in a traffic increase towards stub resolvers of 1.0003x, so the traffic remains almost the same.
System administrators do not usually examine their traffic based on the percentages used above. Commonly queries per second and bytes per second are used (or derived measures). The above fractions are converted by assuming a resolver that services 30000 queries per second. This is a reasonably heavy query load for current resolvers. For resolvers that service a different number of queries per second the traffic can be extrapolated, for a resolver with half the query load, half the traffic would be generated.
So the caching resolver services 30000 queries from stub resolvers per second. Of those 30000 queries, 0.47% result in root lookups, or 141 queries per second. Of those 141 queries, 99.48% need an extra 502 bytes, and the rest an extra 192 bytes. This is 70555 bytes per second extra traffic between the caching resolver and the root servers.
The traffic from stub resolvers that receives answers with DNSSEC data from the root is the nonexistent toplevel domain traffic, this is 4.02% of the 30000. There are 1206 queries per second for nonexistent toplevel domains. For 0.19% of these 1206 queries the DO flag is set, and the answer increases by 502 bytes to accommodate the digital signatures from the root zone. For stub resolvers that do not set the DO flag, nothing changes. This results in 1150 bytes per second extra traffic between the caching resolver and the stub resolvers. If 10% of stub resolvers were to set the DO flag, then 59 kilobytes per second of extra traffic would be generated.

Larger Replies Use More Cache

Larger replies from the root mean that those replies use up more memory space in the cache, and therefore less memory is left for other objects in the cache. This means that the resolver cache hit rate will go down slightly. This in turn will drive up the number of queries towards the root, as the resolver has to query for the information again.
The cache space is shared with information from other DNS servers. Also cache policy keeps often used delegations from the root in cache. Therefore the traffic to the root for negative answers or relatively less common delegations can increase. Mostly traffic to other DNS authority servers may increase as their entries now cache for slightly less long than before.
From the resolver point of view the cache is then a little more in use by root data. The resolver after the root is signed thus operates like a resolver before the root is signed but with a slightly smaller cache memory available. The amount of cache memory lost cannot be bigger than the amount of extra data stored in the root zone compared to the unsigned root zone. This is about 92 kilobytes if signed with a 1024 bit zone signing key and 2048 bit key signing keys.
Assuming the queries are equally popular, an upper bound on the cache size effect can be calculated with ease. The fraction of reduced cache potential for the resolver is the fraction of extra queries that root servers can receive due to this cache size effect. Assuming a resolver with 4 megabytes of cache, the cache space loss can be no larger than 2.24%. Therefore the traffic increase towards all of the authority servers on the internet is no larger than 2.24%.
This fraction is likely to be smaller because relatively unpopular names are removed from the cache to make space for the new root data. The exact value is hard to calculate because it depends on the exact algorithm used by a resolver implementation, and depends on the timing, size and order of arrival of the messages as well as the relative popularity of different domain names. Resolvers with larger caches lose a smaller fraction of cache memory.

Summary

The impact of a signed root on caching resolvers is the increase of several types of traffic. The numbers listed are for a server that has 30 thousand queries per second, results for servers with other query rates can be linearly extrapolated. A server like this already has substantial traffic upstream and downstream before the root is signed in order to provide service to a stream of queries of this magnitude.

The traffic between the stub resolvers and the caching resolver increases by about 1 kilobyte per second. This can go up to 59 kilobytes per second when more customers (10%) update their software to perform DNSSEC validation. The traffic between the caching resolver and the root servers increases by about 69 kilobytes per second. The traffic between the caching resolver and authority servers on the internet other than the root servers increases due to cache limitations and a root size increase, this increase is probably no larger than 2.24%, it may be smaller and it depends on relative query popularity.

The analysis in this document used several assumptions. It was assumed that all EDNS traffic from stub resolvers had the DO flag set. Another assumption was that the root is signed with one 1024-bit RSASHA1 ZSK and there was the assumption that root resource record TTLs remain similar. A completely different assumption was made for the the query trace used: the assumption is that it is representative for the behaviour of other resolvers.

A quick estimate can be extrapolated for larger ZSK key sizes. For a 2048-bit key, 128 bytes more are used per signature. The non-existent toplevel domain answers dominate the outcome. Each has 3 signatures. Add 384 bytes per reply for 1009 bytes on average, a 8.3x increase compared to replies from the unsigned root today.