DNSSEC Operations: Setting the Parameters

SPARTA, Inc.[1]
Shinkuro, Inc.[2]

November 2009

[1] SPARTA, Inc., dba Cobham Analytic Solutions, is supporting the creation of this document under the DNSSEC Deployment Initiative with funding from the Department of Homeland Security Science and Technology (S&T) Directorate under contract FA8750-04-C-0229.

[2] Shinkuro, Inc. produced this document as part of the DNSSEC Deployment Initiative with funding from the Department of Homeland Security Science and Technology (S&T) Directorate under contracts FA8750-04-C-0269 and FA8750-10-C-0020. The information presented does not necessarily represent the views of the U.S. Government. For more information on the initiative, see http://dnssec-deployment.org

shinkuro
シンクロ

Overview

Domain Name System Security Extension (DNSSEC) adds digital signatures to the Domain Name System to secure it against a variety of attacks. With the signing of .ORG and other Top Level Domains (TLDs), registrars that provide name service for their customers and other DNS operators are looking for a reasonable set of DNSSEC configuration parameters. This memo provides advice on the values to choose for the configuration parameters associated with DNSSEC that provide good security without causing an undue burden on operators' name service infrastructures. The configuration parameters include key sizes and lifetimes, re-signing periods, and time-to-live (TTL) for the records. Some of these parameters are visible in the zone; others are internal to the operation.

This is a work in progress. Please provide feedback to dnssec-parameters@shinkuro.com.

Assumptions

We assume that the zones in question are relatively small and neither need nor would benefit from protection against zone walking. This assumption is the basis for some of the parameter values, so it's likely that zones with other attributes (e.g., larger, including non-obvious names) will require different parameters to maintain sufficient security.

All of the configuration parameters are from current experience and are applicable for the near future. Many of the choices are driven by the need to support DNSSEC during the initial phase of adoption. Therefore there will likely be changes as the population (recursive resolvers, clients) supporting DNSSEC grows and as we gain experience dealing with the increased resource requirements (CPU, bandwidth) of longer keys and other variables. In some cases, these choices are at variance with guidance from the U.S. National Institutes and Standards and Technology (NIST). These differences are noted and explained in the text.

DNS settings upon which DNSSEC settings are predicated					
RR Type	TTL				
SOA	1 day				
NS	1 day				
A/AAAA	≤1 day				
DNSKEY	1 day				
Max UDP Packet Size	1492[3]				
SOA Expire Value	1 week				
SOA Negative Cache Time	1 hour				

[3] 4096 is the default max UDP size for many authoritative server implementations. However, PMTU between the authoritative server and the recursive resolver may be less. If there are Ethernet-based links that do not support fragmentation (often due to firewall or similar device), a max UDP size of 1500 or less will be necessary. 1492 includes room for PPoE encapsulation and will suffice for the length of responses likely to be returned based on the DNSSEC parameters suggested above. If the maximum UDP size is exceeded, TCP fallback will be used. Additional measurement using tools like dnsfunnel (http://www.vantage-points.org/) may be needed should TCP fallback be greater than expected.

DNSSEC Settings					
DNSSEC Key Type	RSA(w/SHA1) Key Length[4]	Key Lifetime	Signature Lifetime	Re-Signing	Jitter[5]
KSK	1280	4 years	4 weeks	2 weeks	1 hour
ZSK	1024	1 year	2 weeks	1 week	1 hour
Negative Response	Support				
NSEC	Default				
		Iterations	Salt Size	Salt Lifetime	
NSEC3	Optional[6]	1[7]	64 bits	Signature lifetime	
Rollover	Prepublication/ Signing Policy	Introduction Time[8] for New Key	Retirement Time[9] for Old Key		
KSK	2K,1S[10]	1 week[11]	4 weeks		
ZSK	2K,1S	4 days	2 weeks		

[4] Key lengths are shorter and key lifetimes are longer than current NIST recommendations found in NIST Special Publication 800-81. These values should provide less of a burden in terms of signature size and re-signing in the near term. Since key lifetime is a matter of policy, not protocol, longer keys and shorter lifetimes can be issued in one rollover cycle if it is later determined they are needed.
[5] Jitter represents random variation in the signature timers that prevents certain pre-computation attacks.
[6] See the description NSEC and NSEC3 in the text for more information.
[7] Parameters such as this should be used explicitly in the signing command. BIND 9.6.1, for example, defaults to 100 iterations with NSEC3.
[8] *Introduction Time* is the time a new key of its type (KSK/ZSK) is added before it is the only key being used to sign.
[9] *Retirement Time* is the time an existing key of a type (KSK/ZSK) must continue to be used for signatures after a new key has been introduced.
[10] *2K,1S* means two keys and one active signature. Old keys must be removed to prevent DNSKEY answers from growing in size with each rollover.
[11] Assumes signed parent (TLD). Otherwise, use 45 days per RFC5011.

DS Records

DS records are stored at the parent and tie trust from the parent to the keys that may sign the DNSKEY set in the child. DS records are derived from the subject DNSKEY records per RFC 4034. If a zone is being signed and served by a registrant or a third party, the registrar must accept DS records and pass them to the registry accordingly.

NSEC and NSEC3 Explained

In addition to providing signed responses for resource records that exist, DNSSEC provides signed responses for negative replies (*provable non-existence*). DNSSEC provides a choice of two ways to do this, NSEC and NSEC3. NSEC is simpler to implement and results in smaller packets when a negative answer is required, but it provides an easy way for someone to find out all the names in the zone. NSEC3 provides protection against *zone walking*. Also, for very large zones, NSEC3 reduces the cost of initial operation because NSEC3 records can efficiently facilitate unsigned delegations.

If the zone is not very large and contains names that are easy to guess, e.g. "www," "mail," etc. then NSEC3 provides no advantage. For registrars and others who provide name service for customers with small zones that would not benefit from having their names hidden, we recommend the use of NSEC. For registrars and others who provide name service to customers with large zones or who require the protection of having their names hidden, NSEC3 is necessary.

Underneath the Covers of NSEC and NSEC3

NSEC[12] was the first solution to the problem of providing negative answers and continues to be viable in many cases. At a very high-level, NSEC records identify the gaps in zones. With some level of detail about the types of RRsets available, an NSEC record says little more than there's no name between *A* and *B*. For example, there are no records between mail.example.org and www.example.org. Such an NSEC record can be generated and signed when the rest of the zone is signed and can represent the infinite number of non-existent name/RRset combinations between mail.example.org and www.example.org.

There are two issues with NSEC that caused some to search for an alternative. First, there's *zone walking*. Since NSEC records identify spans of non-existent name/RRsets by identifying the names and RRsets that do exist at the end of the span, it's fairly trivial to fetch one NSEC record after another and *walk* the entire zone, learning every name and RRset it contains. Before the application of DNSSEC, one could prevent the wholesale learning of the contents of a zone merely by disallowing zone transfers. The second issue is that NSEC must specify the status of every delegation (zones below the zone in question) as signed or not. As DNSSEC is rolled out, TLDs like .ORG will be signed before most of the

[12] Defined in RFCs 4034 & 4035

zones below it are signed. Without a more efficient mechanism to identify intent, large zones (e.g., .ORG) will grow larger because even unsigned delegations – the vast majority to start – must have an NSEC record.

Neither of these issues with NSEC will affect a zone with easily guessable names or ones that either do not have child zones or have child zones that are all using DNSSEC.

NSEC3[13] adds a cryptographically hashed name space and a new set of rules to prove non-existence and includes an option that permits unsigned delegations to be omitted from the chain of names protected by NSEC3 records. The former adds privacy when the names in a zone are not otherwise easily guessable and the latter makes a phased rollout of DNSSEC possible. For both of these reasons, some existing TLDs will sign using NSEC3.

NSEC3 requires additional computation when signatures are generated, when negative responses are generated and validated, and on unsigned referrals. NSEC3 negative responses are also larger. These are the reasons we suggest that NSEC be used for the small, simple, guessable, childless zones that many registrars will server for their customers.

Tools Used to Implement DNSSEC

The use of DNSSEC is still relatively new. For those using others' tools, it should be noted that default parameters differ not only between tools, but also between versions of the same tool. Our suggested parameters are not necessarily the default for these tools. Explicit setting of parameters is needed to prevent surprise when changing tools or upgrading existing tools.

[13] Defined in RFC 5155