Workgroup: dnsop
Internet-Draft:
draft-schwartz-dnsop-dnssec-strict-mode-00
Published: 22 February 2021
Intended Status: Standards Track
Expires: 26 August 2021
Authors: B. Schwartz
Google LLC

DNSSEC Strict Mode

Abstract

Currently, the DNSSEC security of a zone is limited by the strength of its weakest signature algorithm. DNSSEC Strict Mode makes zones as secure as their strongest algorithm instead.

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the mailing list (dnsop@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/dnsop/.

Source for this draft and an issue tracker can be found at https://github.com/bemasc/dnssec-strict-mode.

Status of This Memo

Copyright Notice

Table of Contents

1. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Background

2.1. DNSSEC validation behavior

According to [RFC6840] Section 5.4, when validators (i.e. resolvers) are checking DNSSEC signatures:

> a resolver SHOULD accept any valid RRSIG as sufficient, and only determine that an RRset is Bogus if all RRSIGs fail validation.

[RFC6840] Section 5.11 clarifies further:

> Validators SHOULD accept any single valid path. They SHOULD NOT insist that all algorithms signaled in the DS RRset work, and they MUST NOT insist that all algorithms signaled in the DNSKEY

RRset work. A validator MAY have a configuration option to perform a signature completeness test to support troubleshooting.

Thus, validators are required to walk through the set of RRSIGs, checking each one that they are able until they find one that matches or run out.

Some implementations do offer an option to enforce signature completeness, e.g. Unbound's harden-algo-downgrade option [Unbound], but most validating resolvers appear to follow the standards guidance on this point. Validators' tolerance for invalid paths is important due to transient inconsistencies during certain kinds of zone maintenance (e.g. Pre-Publish Key Rollover, [RFC6781] Section 4.1.1.1).

2.2. Algorithm trust levels

From the viewpoint of any single party, each DNSSEC Algorithm (i.e. signature algorithm) can be assigned some level of perceived strength or confidence. The party might be a zone owner, considering which algorithms to use, or a validator, consider which algorithms to implement. Either way, the party can safely include algorithms in which they have maximal confidence (i.e. viewed as secure), and safely exclude algorithms in which they have no confidence (i.e. viewed as worthless).

Under the current DNSSEC validation behavior, a zone is only as secure as the weakest algorithm implemented by both the signer and the validator. If there is at least one algorithm that all parties agree offers maximum strength, this is not a problem. Otherwise, we have a dilemma. Each party is faced with two options:

* Use/implement only their most preferred algorithms, at the cost of achieving no security with counterparties who distrust those algorithms.

* Use/implement a wide range of algorithms, at the cost of weaker security for counterparties who also implement a wide range of algorithms.

In practice, zone owners typically select a small number of algorithms, and validators typically support a wide range. This arrangement often works well, but can fail for a variety of reasons:

* When a new, stronger algorithm is introduced but is not yet widely implemented, zone owners must continue to sign with older, weaker algorithms, typically for many years, until nearly all validators are updated.

* National crypto standards are often highly trusted by some parties, and viewed with suspicion by others.

* Quantum computing has the potential to further confuse the landscape of signature algorithm confidence. Under the present standards, parties might be required to trust a novel postquantum algorithm of uncertain strength or remain vulnerable to quantum attack.

This specification resolves these dilemmas by providing zones with the security level of their strongest selected algorithm, instead of the weakest.

3. The DNSSEC Strict Mode flag

The DNSSEC Strict Mode flag appears in bit $N of the DNSKEY flags field. If this flag is set, all records in the zone MUST be signed correctly under this key's specified Algorithm. A validator that receives a Strict Mode DNSKEY with a supported Algorithm SHOULD reject as Bogus any RRSet that lacks a valid RRSIG with this Algorithm. If there are multiple Strict Mode keys for the zone, validators SHOULD validate signatures under each of their Algorithms.

4. Operational Considerations

Once a zone is signed, enabling Strict Mode can be done using any ordinary key rollover procedure ([RFC6781] Section 4.1), to a new DNSKEY that contains the Strict Mode flag. When signing a zone for the first time, or adding a new Algorithm, care must be taken to fully sign the zone before enabling Strict Mode.

By making it safe to use a wider range of DNSSEC Algorithms, this specification could encourage larger RRSIG RRSets, and hence larger responses.

When a zone has multiple Strict Mode keys, validators will check them all, likely increasing CPU usage.

5. Security Considerations

This specification enables the safe use of signature algorithms with intermediate or indeterminate security. It does not protect against weak Digest Types in DS records (especially "second preimage" attacks).

A zone that adds signatures under a less secure algorithm, relying on a strong Strict Mode algorithm for security, will weaken security for validators that have not implemented support for Strict Mode.

Zone owners should use caution when relying on Strict Mode until Strict Mode is widely supported in validators.

6. IANA Considerations

IANA is instructed to add this allocation to the DNSKEY RR Flags registry:

Number	Description	Reference
$N	STRICT	(This document)

Table 1

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC6840] Weiler, S., Ed. and D. Blacka, Ed., "Clarifications and Implementation Notes for DNS Security (DNSSEC)", RFC 6840, DOI 10.17487/RFC6840, February 2013, <https://www.rfc-editor.org/rfc/rfc6840>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

7.2. Informative References

[RFC6781] Kolkman, O., Mekking, W., and R. Gieben, "DNSSEC Operational Practices, Version 2", RFC 6781, DOI 10.17487/RFC6781, December 2012, <https://www.rfc-editor.org/rfc/rfc6781>.

[Unbound] "unbound.conf", n.d., <https://nlnetlabs.nl/documentation/unbound/unbound.conf/>.

Acknowledgments

TODO acknowledge.

Author's Address

Benjamin M. Schwartz
Google LLC

Email: bemasc@google.com