

DNSOP Working Group
Internet-Draft
Updates: [2308](#), [4033](#), [4034](#), [4035](#) (if
approved)
Intended status: Standards Track
Expires: January 22, 2020

J. Woodworth
D. Ballew
CenturyLink, Inc.
S. Bindiganaveli Raghavan
Hughes Network Systems
D. Lawrence
Oracle
July 21, 2019

Numeric Pattern Normalization (NPN)
draft-woodworth-npn-00

Abstract

The Numeric Pattern Normalization (NPN) resource record provides pre-processing information to reduce the number of possible variants which can be generated by pattern-based RR's into a single signable record.

Ed note

Text inside square brackets ([]) is additional background information, answers to frequently asked questions, general musings, etc. They will be removed before publication. This document is being collaborated on in GitHub at <https://github.com/ionevez/npn>. The most recent version of the document, open issues, etc should all be available here. The authors gratefully accept pull requests.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 22, 2020.

Internet-Draft

NPN

July 2019

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
1.1.	Background and Terminology	3
2.	The NPN Resource Record	3
2.1.	NPN RDATA Wire Format	3
2.2.	The NPN RR Presentation Format	5
2.3.	Use and Normalization Processing of NPN RRs	5
2.3.1.	Pseudocode for NPN Normalization Processing	6
3.	Security Considerations	7
3.1.	Normalized (NPN-Based) Signatures	7
4.	Privacy Considerations	7
5.	IANA Considerations	8
6.	Acknowledgments	8
7.	References	8
7.1.	Normative References	8
7.2.	Informative References	9
Appendix A.	NPN Examples	9
A.1.	EXAMPLE 1	9
A.2.	EXAMPLE 2	10
A.3.	EXAMPLE 3	11
A.4.	EXAMPLE 4	12
	Authors' Addresses	13

[1.](#) Introduction

The Numeric Pattern Normalization (NPN) resource record provides methods to generate DNSSEC signatures for pattern-based "synthesized"

RR's, and validation of such signatures. It does this by introducing a pre-processing step of pattern substitution into both the signing and validating processes.

[1.1.](#) Background and Terminology

The reader is assumed to be familiar with the basic DNS and DNSSEC concepts described in [[RFC1034](#)], [[RFC1035](#)], [[RFC4033](#)], [[RFC4034](#)], and [[RFC4035](#)]; subsequent RFCs that update them in [[RFC2181](#)] and [[RFC2308](#)]; and DNS terms in [[RFC7719](#)].

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 [[RFC2119](#)].

[2.](#) The NPN Resource Record

The Numeric Pattern Normalization (NPN) resource record provides pre-processing information to reduce the number of possible variants that can be generated by a pattern-based RR into one signable record. By identifying parts of the dynamic resource record which should be ignored or represented as a static value, one exemplar record and signature is used to validate all other records that match the pattern.

For example, a pattern replacement like pool-A- $\{1\}$ - $\{2\}$.example.com that generates PTR records for pool-A-0-0.example.com through pool-A-255-255.example.com would have an NPN record that signals a validating resolver to verify all pool-A-#-#.example.com names against a record for pool-A-9-9.example.com.

Though it is conceivable that forged records could be validated as legitimate, such forged records must meet strict criteria and match patterns which define and are considered legitimate. The added measure is a notable improvement in security over being hosted in insecure zones.

The Type value for the NPN RR type is TBD.

The NPN RR is class independent and has no special TTL requirements.

2.1. NPN RDATA Wire Format

The RDATA for an NPN RR consists of a 2 octet Match Type field, a 1 octet Flags field, a 1 octet Owner Ignore field, a 1 octet Left Ignore field and a 1 octet Right Ignore field.

```

      1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|           Match Type           |      Flags      | Owner Ignore |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Left Ignore | Right Ignore |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Match Type indicates the type of the RRset with which this record is associated.

Flags defines additional processing parameters for data normalization. This document defines only the Period-As-Number flag "." (position 5), the Hyphen-As-Number "-" (position 6) and the hexadecimal flag "X" (position 7). All other flags are reserved for future use.

```

0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|Reserved|.|-|X|
+---+---+---+---+---+---+
```

Bits 0-4: Reserved for future

Bit 5: Period As Number (.) Flag

If 0, periods are treated as non-digits.

If 1, periods will be processed as digits.

Bit 6: Hyphen As Number (-) Flag

If 0, hyphens are treated as non-digits.
If 1, hyphens will be processed as digits.

Bit 7: Hexadecimal (X) Flag

If 0, numeric digits include only 0-9.

If 1, numeric digits include a-f in addition to 0-9.

Owner Ignore defines the number of octets in the owner name, as counted from the left, which MUST be ignored by the normalization process. This field offers additional security to pattern based signatures which may not be immediately apparent. By restricting the leftmost characters defined by this value, ultimately the length of the generated portion of the accompanying synthesized RR will be confined accordingly.

Left Ignore defines the number of octets of the generated RDATA, as counted from the left, which MUST be ignored by the normalization process.

Right Ignore defines the number of octets of the generated RDATA, as counted from the right, which MUST be ignored by the normalization process.

[2.2.](#) The NPN RR Presentation Format

Match Type is represented as an RR type mnemonic or with [\[RFC3597\]](#)'s generic TYPE mechanism.

Flags is a string of characters indicating the status of each bit as per the following table. The characters can appear in any order.

+-----+	+-----+	+-----+
Flag	Unset	Set
+-----+	+-----+	+-----+
Period As Number		.
+-----+	+-----+	+-----+
Hyphen As Number		-
+-----+	+-----+	+-----+
Hexadecimal	9	f
+-----+	+-----+	+-----+

Owner Ignore, Left Ignore, and Right Ignore are displayed as unsigned decimal integers, ranging from 0 through 255.

[2.3.](#) Use and Normalization Processing of NPN RRs

This document provides a minor yet significant modification to DNSSEC regarding how RRsets will be signed or verified. Specifically the Signature Field of [\[RFC4034\], Section 3.1.8](#). Prior to processing into canonical form, signed zones may contain associated RRs where; owner, class and type of a non NPN RR directly corresponds with an NPN RR matching owner, class and Match Type. If this condition exists the NPN RR's RDATA defines details for processing the associated RDATA into a "Normalized" format. Normalized data is based on pre-canonical formatting and zero padded for "A" and "AAAA" RR types for acceptable precision during the process. This concept will become clearer in the NPN pseudocode and examples provided in the sections to follow.

The rules for this transformation are simple:

- o For RR's Owner field, characters from the beginning to the index of the Owner Ignore value or the final string of characters belonging to the zone's ORIGIN MUST NOT be modified by this algorithm. This value is intended to provide additional limitations on the query portion further minimizing the risk of spoofed RR's matching NPN-based signatures.

- o For RR's RDATA field, character from beginning to the index of Left Ignore value or characters with index of Right Ignore value to the end MUST NOT be modified by this algorithm.
- o In the remaining portion of both Owner and RDATA strings of numeric data, defined as character "0" through "f" or "0" through "9" depending on whether or not the Hexadecimal flag is set or not, MUST be consolidated to a single character and set to the highest value defined by the Hexadecimal flag. Examples may be found in the following section. If period-as-number or hyphen-as-number flags are set whichever are used ("." or "-") would be treated as part of the number and consolidated where appropriate.

Once the normalization has been performed the signature will continue processing into canonical form using the normalized RRs in the place

of original ones.

If multiple NPN RR's resolve to the same owner and type, it MUST be assumed either multiple overlapping pattern-based generation RR's coexist for the same owner. In this scenario, the validation algorithm SHOULD be attempted against all NPN RR's until a successful validation is found or no NPN's for this owner and type remain. While multiple overlapping pattern-based generation SHOULD be discouraged, future pattern-based RR's may necessitate this condition so it must be accounted for.

NPN RRs MAY be included in the "Additional" section to provide a hint of the NPN processing required for the verification path.

It is important to note, properly sizing the Ignore fields is critical to minimizing the risk of spoofed signatures. Never intentionally set all Ignore values to zero in order to make validation easier as it places the validity of zone data at risk. Only accompany RRs which are pattern derived (such as BULK) with NPN records as doing so may unnecessarily reduce the confidence level of generated signatures.

[2.3.1](#). Pseudocode for NPN Normalization Processing

This section provides a simple demonstration of process flow for NPN rdata normalization and DNSSEC signatures.

The pseudocode provided below assumes all associated RRs are valid members of a DNSSEC-compatible RRset, including pattern-generated ones.

```
for rr in rrset
    if (has_NPN<rr.owner, rr.class, rr.type>)
        rr.rdata_normal = NPN_normalize<rr.rdata>
        add_to_sigrrset<NPN.owner, rr.class, rr.type,
            rr.rdata_normal>
    next
else
    add_to_sigrrset<rr.owner, rr.class, rr.type, rr.rdata>
```

next

process_canonical_form<sigrrset>

dnssec_sign<sigrrset>

Similar logic MUST be used for determining DNSSEC validity of RRsets in validating nameservers for signatures generated based on NPN normalization.

[3.](#) Security Considerations

The NPN RR is intended to be used only with the offline-signing of pattern-based RR's where there is an expected minimal security impact. Use of NPN RR's for other purposes is strongly discouraged and falls outside the scope of this document.

[3.1.](#) Normalized (NPN-Based) Signatures

This solution provides a flexible solution as nameservers without on-the-fly signing capabilities can still support signatures for pattern-based records. The down side to this solution is it requires NPN resolver support for validation.

It has been pointed out that due to this limitation, creation of DNSSEC-signed pattern-based RRs requiring NPN support SHOULD be formally discouraged until such time a respectable percentage (>80%) of validating resolvers in-the-wild possess NPN processing capabilities. Until that time, on-the-fly signing and unsigned records offer the intended capabilities for pattern-based RR's, while requiring zero new features to support their resolution. Given enough time, enough nameservers will be patched and upgraded for unrelated reasons and by means of simple attrition can supply a level of inertia and eventually widespread adoption can be assumed.

[4.](#) Privacy Considerations

NPN records do not introduce any new privacy concerns to DNS data.

[5.](#) IANA Considerations

IANA is requested to assign numbers for the NPN DNS resource record type identified in this document.

6. Acknowledgments

This document was created as an extension to the DNS infrastructure. As such, many people over the years have contributed to its creation and the authors are appreciative to each of them even if not thanked or identified individually.

A special thanks is extended for the kindness, wisdom and technical advice of Robert Whelton (CenturyLink, Inc.) and Gary O'Brien (Secure64).

7. References

7.1. Normative References

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, [RFC 1034](#), DOI 10.17487/RFC1034, November 1987, <<https://www.rfc-editor.org/info/rfc1034>>.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, [RFC 1035](#), DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.
- [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/info/rfc2119>>.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", [RFC 2181](#), DOI 10.17487/RFC2181, July 1997, <<https://www.rfc-editor.org/info/rfc2181>>.
- [RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS NCACHE)", [RFC 2308](#), DOI 10.17487/RFC2308, March 1998, <<https://www.rfc-editor.org/info/rfc2308>>.
- [RFC2317] Eidnes, H., de Groot, G., and P. Vixie, "Classless IN-ADDR.ARPA delegation", [BCP 20](#), [RFC 2317](#), DOI 10.17487/RFC2317, March 1998, <<https://www.rfc-editor.org/info/rfc2317>>.

- [RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", [RFC 3597](#), DOI 10.17487/RFC3597, September 2003, <<https://www.rfc-editor.org/info/rfc3597>>.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", [RFC 4033](#), DOI 10.17487/RFC4033, March 2005, <<https://www.rfc-editor.org/info/rfc4033>>.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", [RFC 4034](#), DOI 10.17487/RFC4034, March 2005, <<https://www.rfc-editor.org/info/rfc4034>>.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", [RFC 4035](#), DOI 10.17487/RFC4035, March 2005, <<https://www.rfc-editor.org/info/rfc4035>>.

[7.2.](#) Informative References

- [RFC7719] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", [RFC 7719](#), DOI 10.17487/RFC7719, December 2015, <<https://www.rfc-editor.org/info/rfc7719>>.

[Appendix A.](#) NPN Examples

[A.1.](#) EXAMPLE 1

```
2.10.in-addr.arpa. 86400 IN BULK PTR (
                                [0-255].[0-10].2.10.in-addr.arpa.
                                pool-A- $\{1\}$ - $\{2\}$ .example.com.
                                )
2.10.in-addr.arpa. 86400 IN NPN PTR 9 0 7 13
```

For the BULK RR used in this example, a query for the PTR of address 10.2.3.44 would match the for the query name 44.3.2.10.in-addr.arpa, resulting in the generation of a PTR to pool-A-3-44.example.com as the response.

By protecting the "Ignore" characters from the generated RR's RDATA the focus for normalization becomes "3-44" as illustrated below.

Internet-Draft

NPN

July 2019

```

0 1 2 3 4 5 6 7
          v-----
p o o l - A - 3 - 4 4 . e x a m p l e . c o m .
          -----^
                        1 1 1 1
                        3 2 1 0 9 8 7 6 5 4 3 2 1 0

```

Everything to the left of "3-44" will remain intact, as will everything to its right. The remaining characters will be processed for digits between 0 and 9 as indicated by the NPN record's hexadecimal flag 9, and each run of digits replaced by the single character "9". The final Normalized RDATA for *.2.10.in-addr.arpa would therefore become pool-A-9-9.example.com., and its signature would be based on this value to cover all possible permutations of the pool-A- $\{1\}$ - $\{2\}$.example.com replacement pattern.

Since the validating nameserver would use the identical NPN record for processing and comparison, all RRs generated by the BULK record can now be verified with a single signature.

[A.2.](#) EXAMPLE 2

This example contains a classless IPv4 delegation on the /22 CIDR boundary as defined by [RFC2317](#). The network for this example is "10.2.0/22" delegated to a nameserver "ns1.sub.example.com.". RRs for this example are defined as:

```

$ORIGIN 2.10.in-addr.arpa.
0-3 86400 IN      NS      ns1.sub.example.com.
      86400 IN BULK CNAME [0-255].[0-3] ${*|.}.0-3
      86400 IN NPN  CNAME 9 0 0 23

```

For this example, a query of "10.2.2.65" would enter the nameserver as "65.2.2.10.in-addr.arpa." and a "CNAME" RR with the RDATA of "65.2.0-3.2.10.in-addr.arpa." would be generated.

By protecting the "Ignore" characters from the generated RR's RDATA the focus for normalization becomes "65.2" as illustrated below.

```

0
V-----
 6 5 . 2 . 0 - 3 . 2 . 1 0 . i n - a d d r . a r p a .
-----^
          2 2 2 2 1 1 1 1 1 1 1 1 1 1 1
          3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0

```

Everything to the left of "65.2" will remain intact as will everything to its right. The remaining characters will be processed

Woodworth, et al. Expires January 22, 2020 [Page 10]

Internet-Draft NPN July 2019

for digits from 0 through 9 as indicated by the NPN record's hexadecimal flag "9", with each run replaced by the single character "9". The final normalized RDATA would therefore become 9.9.0-3.2.10.in-addr.arpa and its signature would be based on this normalized RDATA field. This new normalized string would be used as an RDATA for the wildcard label of 2.10.in-addr.arpa now encompassing all possible permutations of the `${*|.}.0-3.2.10.in-addr.arpa` pattern.

As in example 1, the validating resolver would use the same NPN record for comparison.

[A.3.](#) EXAMPLE 3

This example provides reverse logic for example 1 by providing an IPv4 address record for a requested hostname. For this example the query is defined as an A record for `pool-A-3-44.example.com`, with an origin of `example.com`. RRs for this example are defined as:

```

example.com. 86400 IN BULK A (
                                pool-A-[0-10]-[0-255]
                                10.2.${*}
                                )
example.com. 86400 IN NPN  A 9 0 8 0

```

By protecting the "Ignore" characters from the generated RR's RDATA the focus for normalization becomes "003.044" as illustrated below.

```

0 1 2 3 4 5 6 7 8
          V-----
0 1 0 . 0 0 2 . 0 0 3 . 0 4 4
          -----^

```

This example illustrates a key point about NPN records; since they are pre-canonical they MUST operate on a strict subset of WIRE formatted data. For A and AAAA records this means the "Ignore" fields are based on zero padded data. In this example our generated record MUST be converted into "010.002.003.044" (shown above) prior to processing. After processing, wire format would become "0x0A02032C" (shown in hexadecimal). This format would be too imprecise for normalization so padded decimal is used.

Everything to the left of "003.044" will remain intact as will everything to its right. The remaining characters will be processed for digits between 0 and 9 as indicated by the NPN record's hexadecimal flag "9" and each run replaced by the single character "9". The final Normalized RDATA would therefore become 10.2.9.9 and

its signature would be based on this normalized RDATA field. This new normalized A RR would be used as an RDATA for the wildcard label of "*.example.com." now encompassing all possible permutations of the 10.2.\${*} pattern.

[A.4.](#) EXAMPLE 4

This example provides similar logic for an IPv6 AAAA record. For this example the query is defined as an AAAA record for pool-A-ff-aa.example.com with an origin of example.com.. RRs for this example are defined as:

```
example.com. 86400 IN BULK AAAA (
                                pool-A-[0-ffff]-[0-ffff]
                                fc00::${1}:${2}
                                )
example.com. 86400 IN NPN  AAAA X 0 30 0
```

By protecting the "Ignore" characters from the generated RR's RDATA the focus for normalization becomes "00ff:00aa" as illustrated below.

```

      1 1 1 1 1 1 1 1 1 1 2 2
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

f c 0 0 : 0 0 0 0 : 0 0 0 0 : 0 0 0 0 : -/-/
```

```

4 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 1
0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9
/-/-/- . . . . . . . . . . . . . . . . . . . -/-/-/
                2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4
                1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
                                v-----
/-/- 0 0 0 0 : 0 0 0 0 : 0 0 f f : 0 0 a a
                                -----^
                2 1 1 1 1 1 1 1 1 1 1
                0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0

```

This example reinforces the point on pre-canonical processing of NPN records; they MUST operate on a strict subset of WIRE formatted data. For A and AAAA records this means the "Ignore" fields are based on zero padded data. In this example our generated record MUST be converted into "fc00:0000:0000:0000:0000:0000:00ff:00aa" (shown above) prior to processing. After processing, wire format would become "0xFC0000000000000000000000FF00AA" (shown in hexadecimal). This format is slightly misleading as it is truly only 16 bytes of WIRE data and would be too imprecise for normalization so padded hexadecimal is used.

Everything to the left of "00ff:00aa" will remain intact as will everything to its right. The remaining characters will be processed for numbers between "0" and "f" as indicated by the NPN record's hexadecimal flag "X" and each run replaced by the single character "f". The final Normalized RDATA would therefore become "fc00::f:f" and its signature would be based on this "normalized" RDATA field. This new "normalized" "AAAA" RR would be used as an RDATA for the wildcard label of *.example.com now encompassing all possible permutations of the "fc00::\${1}:\${2}" pattern.

Authors' Addresses

John Woodworth
 CenturyLink, Inc.
 4250 N Fairfax Dr
 Arlington VA 22203
 USA

Email: John.Woodworth@CenturyLink.com

Dean Ballew
CenturyLink, Inc.
2355 Dulles Corner Blvd, Ste 200 300
Herndon VA 20171
USA

Email: Dean.Ballew@CenturyLink.com

Shashwath Bindiganaveli Raghavan
Hughes Network Systems
11717 Exploration Lane
Germantown MD 20876
USA

Email: shashwath.bindiganaveliraghavan@hughes.com

David C Lawrence
Oracle

Email: tale@dd.org