あらためて、TTLについて

やまぐちたかのり

2025/11/26 DNSPOS.JP BoF

TTLって?

```
www.example.com. 86400 IN A 192.0.2.1
```

↑これ

- みなさんご存知、キャッシュの生存時間
- ……ほんとにご存知?

RRsetのTTL

- 書式上は、リソースレコードごとに異なるTTLを設定できる

```
www.example.com.     60 IN A 192.0.2.1 ; 1m
                   3600 IN A 192.0.2.2 ; 1h
                  86400 IN A 192.0.2.3 ; 1d
```

- RFC2181曰く「RRset内のすべてのRRは同一TTLでなければならない」
  - RRset: 同じ名前で同じタイプのリソースレコードをひとまとめにしたもの
  - つまり、上の例は正しくない

異なるTTLを含むRRsetの扱い

- RFC2181 section 5.2 (JPRSさんの日本語訳)

RRSetに属するRRが異なるTTLを持つ応答をクライアントが受信した場合、エラーとして処理すべきである。当該RRSetが権威を持たない情報源から取得したものである場合、クライアントは単にRRSetを無視すべきである。その情報が必要ならば、権威を持つ情報源からの取得を試みるべきである。すべての問い合わせを一つ以上の特定のサーバーに送信するように設定されたクライアントは、この目的の場合には、これらのサーバーを権威を持つ情報源と扱うべきである。

権威サーバーが異なるTTLを持つ不正なRRSetを送信する場合、クライアントは、RRSetに属するすべてのRRのTTLに、RRSet内のTTL最小値が設定されているものとして処理すべきである。いかなる場合も、サーバーが異なるTTLを持つRRSetを送信してはならない。

https://jprs.jp/tech/material/rfc/RFC2181-ja.txt

←エラーとして扱え

←最小値をTTLとみなせ

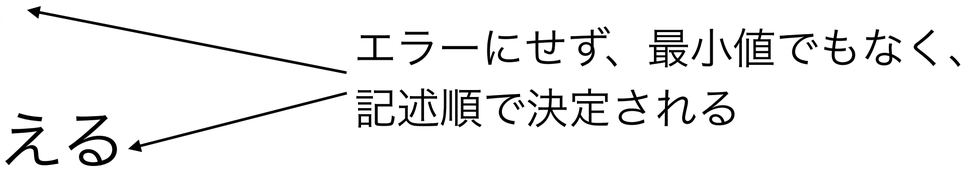
どっちなんだよ

試してみた

- 以下のRRsetを含むゾーンを読み込ませてみる

```
www.example.com.     60 IN A 192.0.2.1 ; 1m
                   3600 IN A 192.0.2.2 ; 1h
                  86400 IN A 192.0.2.3 ; 1d
```

- BIND: 先頭のRRのTTLを採用して60秒に揃える
- Knot: 最後のRRのTTLを採用して86400秒に揃える
- NSD: エラーにする
- PowerDNS: 異なるTTLのまま受け入れて異なるTTLのまま応答する



PowerDNSくん…

```
% dig +norec +noall +ans @127.0.0.1 www.example.com
www.example.com.        3600    IN      A       192.0.2.2
www.example.com.        60      IN      A       192.0.2.1
www.example.com.        86400   IN      A       192.0.2.3
% dig +short @127.0.0.1 ch txt version.bind
"PowerDNS Authoritative Server 5.1.0-
alpha0.676.master.g611faa4b0 (built Oct 21 2025 04:41:25 by
root@localhost)"
```

- このTTLをどう解釈するかはフルリゾルバの実装依存

想定されるトラブル(1)

- BINDを捨てて別の権威サーバ実装に乗り換えよう

```
www.example.com.     60 IN A 192.0.2.1
                   3600 IN A 192.0.2.2
```

- 乗り換え先がKnotの場合: TTLが60秒→3600秒に
- 乗り換え先がNSDの場合: BINDで使えていたゾーンファイルを読み込めない
- 乗り換え先がPowerDNSの場合: フルリゾルバによってTTLが60秒になったり3600秒になったり

想定されるトラブル(2)

```
www.example.com. 3600 IN A 192.0.2.1
                 3600 IN A 192.0.2.2
```

- 192.0.2.2のホストをリプレースするので事前にTTLを短縮しておこう

```
www.example.com. 3600 IN A 192.0.2.1
                   60 IN A 192.0.2.2
```

- BINDでは先頭のTTL(=86400)が採用されるので、短縮したことにならない
- 両方のRRのTTL変更が必要

まとめ

- **RRsetを構成するRRのTTLは同じでなければならない**
  - 基本的かつ重要なことなんだけど、この件に関する言及って驚くほど少ないんですよ (JPRSさんの『DNSがよくわかる教科書』にも書いてない)
- 異なっていた場合の扱いは実装依存
  - ゾーンを記述する側が意識して揃えておかないとトラブルになりうる
  - で、結局のところRFC2181 #5.2の正しい解釈は?

- この先は当日時間があれば

セカンダリサーバの挙動

- プライマリからのゾーン転送で得られたRRsetが異なるTTLだった場合
- BIND: 転送されてきたRRsetの先頭のTTLに値を揃える
- Knot: 転送されてきたRRsetの最後のTTLに値を揃える
- NSD、PowerDNS: 異なるTTLのまま受け入れて異なるTTLのまま応答する
- NSD以外はプライマリでのゾーン読み込みと同じ挙動

フルリゾルバの挙動

- 権威サーバから異なるTTLを持つRRsetが返ってきた場合
- BIND: 最小値に揃えて応答し、そのTTLが過ぎるとキャッシュから消える
- Unbound、PowerDNS Recursor: 異なるTTLのまま応答するが、最小のTTLが過ぎるとRRsetごとキャッシュから消える
- Knot Resolver: いずれかのTTL(どれが採用されるかは不定)に揃えて応答し、そのTTLが過ぎるとキャッシュから消える
- いずれもエラーにはしない

RRSIGのTTL (1)

- RRSIGはTTLが揃ってなくてもよい

```
example.com.  86400 IN SOA    ……
              86400 IN NS     ……
                600 IN A      ……
              86400 IN RRSIG  SOA 13 2 86400 ……
              86400 IN RRSIG  NS  13 2 86400 ……
                600 IN RRSIG  A   13 2 600   ……
```

- 「RRSIG RRは署名対象のRRsetと強く結びついているため、他のDNS RRタイプと異なりRRsetを形成しない..(略)..複数のRRSIG RRのTTL値は[RFC2181]が規定する規則に従わない」(RFC4035 #2.2 JPRSさん訳)

RRSIGのTTL (2)

- RRSIGのTTLは署名対象RRsetのTTLと同一でなければならない
- じゃあ、RRset内のRRのTTLが異なることがあるPowerDNSで署名すると?

RRSIG自身のTTL (フルリゾルバでカウントダウンされる)

```
www.example.com.  86400 IN  RRSIG A 13 3 86400 ……
www.example.com.  3600  IN  A     192.0.2.2
www.example.com.  60    IN  A     192.0.2.1
www.example.com.  86400 IN  A     192.0.2.3
```

original TTL (署名対象RRsetのTTL; カウントダウンされない)

- バリデータの挙動: 検証成功(bind, unbound, knot-res, pdns-rec)