DNSでHTTPS

※ただしDNS over HTTPSではない

DNS Summer Day 2021
山口崇徳@IIJ

HTTP URL

```
https://www.example.com/
```

- www.example.comというホストにHTTPSというスキームでアクセスしてね

- URLに示されるホスト名 = 実際にアクセスされるホスト名
- あえて変えたい場合は → CNAME

CNAME

```
www.example.com. IN CNAME example.com.
```

- www.example.comというホストの正規名はexample.comだよ
  - 別名(alias)ではなく、正規名(canonical name)
- CNAMEの制限
  - 他のレコードとの同居不可
    - ゾーン頂点(ゾーン名自身)にはかならずSOAとNSが存在する → ゾーン頂点でのCNAME不可
  - 特定プロトコルだけを選択的にCNAMEの対象にできない

MX

`foo@example.com`

- example.comドメインのfooさん
- example.comドメイン宛のメールをexample.comというホストが受けとるわけではない

```
example.com. IN MX 10 mail1.example.com.
             IN MX 20 mail2.example.com.
```

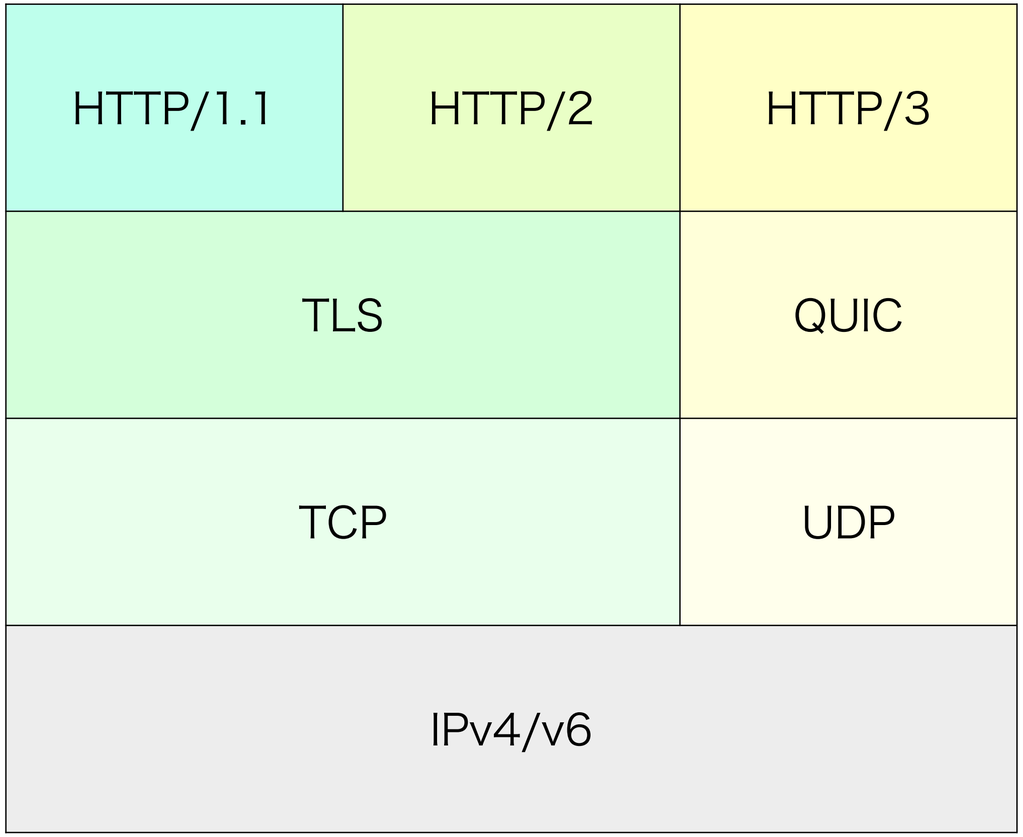
  - mail1で受信、そのホストがコケてたらかわりにmail2が
- メール以外では?

SRV

```
_foo._tcp.example.com. IN SRV (
                1 10 1234 foo.example.com. )
```

- example.comドメインのfooというサービスにTCPでアクセスするには、foo.example.comというホストの1234ポートを使ってね
  - 優先度設定に加えて重みづけ負荷分散もできる (この例ではpriority 1、weight 10)
- CNAMEのような制限がなく、MXのように特定プロトコル限定でない
  - が、HTTPでは使わない
  - HTTP/2がまだドラフトだったころに採用する案はあったようだけど

HTTPバージョン



- HTTPの各バージョンに互換性はないため、バージョン選択の仕組みが必要
- HTTP/1.1、/2: TLSの層でネゴシエーション(ALPN)
- HTTP/3: TLSより下の層から異なるため、アクセス後のネゴは不可
  - HTTP/1.1、/2の接続中にHTTP/3対応を通知(AltSvc) → 次回接続からHTTP/3に
  - 事前知識があれば初回からHTTP/3可能

Encrypted Client Hello

- SNI: TLSハンドシェイク中、クライアントがアクセスしたいホスト名をサービスに伝える仕組み
  - 従来のTLSでは暗号化が開始される前に平文でSNIのやりとり
  - 特定のSNIだけを選択的に中間者攻撃できてしまう
- Encrypted Client Hello (ECH)
  - https://datatracker.ietf.org/doc/draft-ietf-tls-esni/
  - SNIなどClientHelloで送られる各種情報を暗号化するTLS1.3拡張
  - ハンドシェイクの最初から暗号化するので、必要な公開鍵は事前に入手しておく必要がある

背景まとめ

- メールはドメイン名と異なるホスト名で受け取れるが、HTTPはURLに示されたホスト名がアクセスを受けなければならない
- CNAMEを使えば異なるホスト名で処理することもできるが、すべてのケースで使えるとはかぎらない
- HTTPSで接続する際に事前に知っておいた方が有利な情報(HTTPバージョン、ECH鍵)はSRVに書けない

- こういう問題を一気に解決したい

draft-ietf-dnsop-svcb-https

- こういった問題を解決するための新しいリソースレコードを定義するぜ! というInternet draft
  - https://datatracker.ietf.org/doc/draft-ietf-dnsop-svcb-https/
  - Web屋さんが積極的に推進
  - DNS屋さんも好意的に受けとめ
- dnsop wgでの議論はすでに終了
  - 現時点での最新版(-06)はWGLCでの意見を反映したもので、これがおそらく最終版
  - 体裁だけ整えた後でRFCになると思われる

SVCB/HTTPS RR

- ドラフトで新たに定義される2種類のリソースレコード
  - “SVCB” (service binding): 汎用
  - “HTTPS”: HTTP専用
  - IANA登録済み(Type64/65)
- 雑に言うと、CNAMEとMXを足して2で割らずに福神漬を添えたもの
- HTTPSというネーミングはどうにかならんかったのか…
  - 初期のドラフトでは“HTTPSSVC”だった
  - まぎらわしい文脈では“HTTPS RR”と呼ぶことにします

つまり、どういうこと?

- WebサーバのDNSへの登録方法が変わるってこと

https://www.example.com/

- これまで

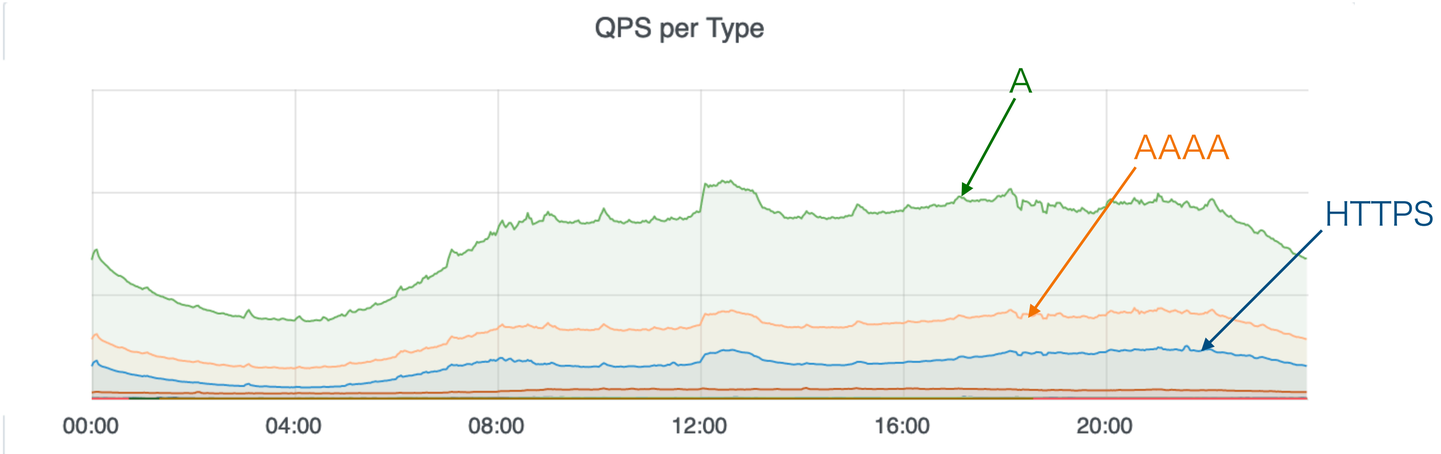
```
www.example.com.  IN A      192.0.2.1          ; アクセスされるIPアドレスを設定
                  IN AAAA   2001:db8::1
```

- これから


```
www.example.com.  IN HTTPS  1 svr.example.com. ; アクセスされるホスト名を設定
svr.example.com.  IN A      192.0.2.1          ; そのホスト名のIPアドレスを設定
                  IN AAAA   2001:db8::1
```

HTTPS RRの利用状況

- 現時点ですでに全クエリの中でA、AAAAに次いで3番目に多い
  - IIJのキャッシュサーバでは全体の1割以上がHTTPS RR (2021/06時点)



  - Akamaiの調査
    - https://indico.dns-oarc.net/event/37/contributions/810/attachments/784/1413/dns-https-rr-final.pdf

HTTPS RRの実装状況

- Webブラウザ
  - Chrome: 対応済みのはず (手元では設定しても使ってくれない…)
  - Firefox: 対応済み (設定変更が必要)
  - Safari: 対応済み (設定不要で使いまくっている)
- オープンソースなDNSサーバでは、現状ではPowerDNSのみが対応
  - それ以外は開発中だが、どれも早めに対応版がリリースされそうな雰囲気
- WebとDNSの両方がCloudflareにホスティングされているサイトの多く(すべて?)でHTTPS RRが登録されている模様

書式

```
name IN SVCB|HTTPS SvcPriority TargetName [SvcParams]
```

- SvcPriory: 0-65535の整数(必須)
  - 0: エイリアスモード
  - 0以外: サービスモード

- TargetName: 処理の委任先(必須)
- SvcParams: key=valueペアのリスト(オプション)
  - エイリアスモード: 無視される
  - サービスモード: 構成情報の定義

エイリアスモード

- 優先度0のHTTPS RRはエイリアスモード

```
example.com. IN HTTPS 0 svr.example.net.
```

- シンプルなエイリアスを実現するモード
  - クライアントはターゲット名(この例ではsvr.example.net)のHTTPS RRを再度問い合わせる
    - 多段エイリアスが可能
    - CNAME→エイリアスモード、エイリアスモード→CNAMEという多段構成も可
- 雑に言うと、HTTPS限定でゾーン頂点でも使えるCNAME

サービスモード

- 優先度0以外のHTTPS RRはサービスモード

```
svr.example.net.  IN HTTPS 10 svr1.example.net. alpn=h3,h2 ech="abc…"
                  IN HTTPS 20 svr2.example.net. alpn=h2 ech="123…"
```

- 実際のサービス提供ホストを提示するモード
  - クライアントはターゲット名のA/AAAA RRを問い合わせて接続先を決定する
  - 優先度の値が小さい順に、値が同じならランダムにアクセス(MXと同じ)
  - アクセスに必要な情報を書ける
- 雑に言うと、追加情報を書けるHTTP用のMX

サービスパラメータ(1)

```
example.com. IN HTTPS 10 svr.example.net. alpn=h3,h2 ech="abc…"
```

- サービスモードでのみ書けるパラメータ
  - エイリアスモードでは無視される(書いてもエラーにはならない)
- alpn、no-default-alpn
  - サポートしているプロトコル
  - http/1.1はデフォルトなので記述しなくてもよい
- ech
  - Encrypted Client Helloの鍵情報(ECHConfigList)

サービスパラメータ(2)

- port
  - 非標準ポート番号の利用
- ipv4hint、ipv6hint
  - IPアドレス (A/AAAAで得られた情報がある場合はそちらが優先)
- mandatory
  - サービスパラメータが “foo=bar mandaroty=foo” という指定だった場合、
    - fooという機能に対応しているクライアントのみがそのサービスにアクセスする
    - fooに対応していないクライアントはこのレコードを無視する

典型的な利用例

```
https://example.com/
```

```
example.com. IN HTTPS 0 cdn.example.net.
```

URLに出現する名前にCDNサービスへのエイリアスを設定

CDN側はサービスモードでサーバの詳細を記述

```
cdn.example.net.   IN HTTPS 10 sv30.example.net. alpn=h3,h2
                   IN HTTPS 10 sv31 example.net. alpn=h3,h2
                   IN HTTPS 20 sv20.example.net. alpn=h2
sv30.example.net. IN A     192.0.2.1
sv31.example.net. IN A     192.0.2.2
sv20.example.net. IN A     192.0.2.3
```

“ ”
■

- DNSの世界で “.” は一般にルートドメインだが、HTTPS RRでは別の意味

`example.com. IN HTTPS 0 .`

- エイリアスモードでは、example.comが利用できないことを示す

`example.com. IN HTTPS 1 .`

- サービスモードでは、ターゲットが自分自身であることを示す
- これの省略形 → `example.com. IN HTTPS 1 example.com.`

平文HTTPでのHTTPS RR

- HTTPS RRは http:// なURLでも参照される
  - その結果有効なHTTPS RRが見つかれば、https:// にリダイレクトする
- すなわち、HSTSのもうひとつの実現方法
  - レスポンスのStrict-Transport-Securityヘッダによる指示
    - HTTPSになるのは2回目のアクセスから
  - ブラウザ組み込みのHTTPS強制ドメインのリストに含まれるサイト(HSTS preload)
    - 自分のドメインをリストに入れてもらうよう申請が必要
  - HTTPS RR ← new!
    - DNS登録だけで初回アクセスからHTTPSに

セキュリティ

- 通信経路上の中間者はHTTPS RRの名前解決を失敗させる/応答を改竄することでダウングレード攻撃が可能
  - ECH公開鍵が得られず平文SNIに → SNIブロッキング
  - HTTPS RRによるHSTSが効かない → 平文HTTP盗聴
- このような攻撃を検知した場合、クライアントは接続をやめるべき(SHOULD)
  - 仕様上はSVCB/HTTPS RRの利用にDNSSEC/暗号化DNSは必須ではない
  - が、これを考慮して(?)、Firefox、Chromeは現状ではDoH必須
    - 将来的にもDoH必須とするのかどうかは不明

SVCB RR

- HTTPS RRはHTTPとその派生プロトコル(HTTPS、WebSocket)用
- SVCB RRはプロトコルを限定しない
  - scheme://name[:port] というURLに [_port.]_scheme.name のSVCB RRが対応
  - つまり、以下の2つは意味的には同じ

```
example.com.          IN HTTPS …

_https.example.com. IN SVCB  …
```

    - ただし、https:// なURLではかならずHTTPS RRを使うことになっていて、SVCBは無効
- https:// 以外に使われること、qnameが異なること以外はHTTPSとほぼ同じ

SVCBで暗号化DNS(1)

- Discovery of Designated Resolvers (draft-ietf-add-ddr)
  - DoH/DoTを使うために必要な情報を取得するための手順を定義するドラフト
  - 以下の名前のSVCB RRを問い合わせてDoH/DoTサーバの情報を得る
    - Do53フルリゾルバのIPアドレスだけがわかっている場合: _dns.resolver.arpa
    - DoH/DoTサーバの名前はわかっている場合: _dns.{DoH/DoTサーバ名}
  - こんなSVCB RRを登録する

```
_dns.example.com. IN SVCB 1 . alpn=h2 dohpath=/dns-query{?dns} ipv4hint=x.y.z.w
_dns.example.com. IN SVCB 1 . alpn=dot port=8530 ipv4hint=x.y.z.w
```

SVCBで暗号化DNS(2)

- draft-ietf-dprive-unauth-to-authoritative
  - フルリゾルバが権威サーバにDoT/DoQなどでアクセスする手順を定義するドラフト
  - 議論が始まったばかりでまだ具体的な記述は少ないものの、SVCB RRを利用する想定になっている

- ちうことで、HTTPS RRだけでなくSVCB RRの方も具体的な利用の提案がはじまっている
  - 実際にRFCとして標準化されるかどうかは現時点ではなんとも…

オーバーヘッド大きくない?

- これまで
  - A/AAAAの問い合わせだけ
- これから
  - HTTPS RRの問い合わせ + A/AAAAの問い合わせ
- HTTPS RRの分だけDNSの問い合わせが増える → レイテンシ低下?

レイテンシ向上策

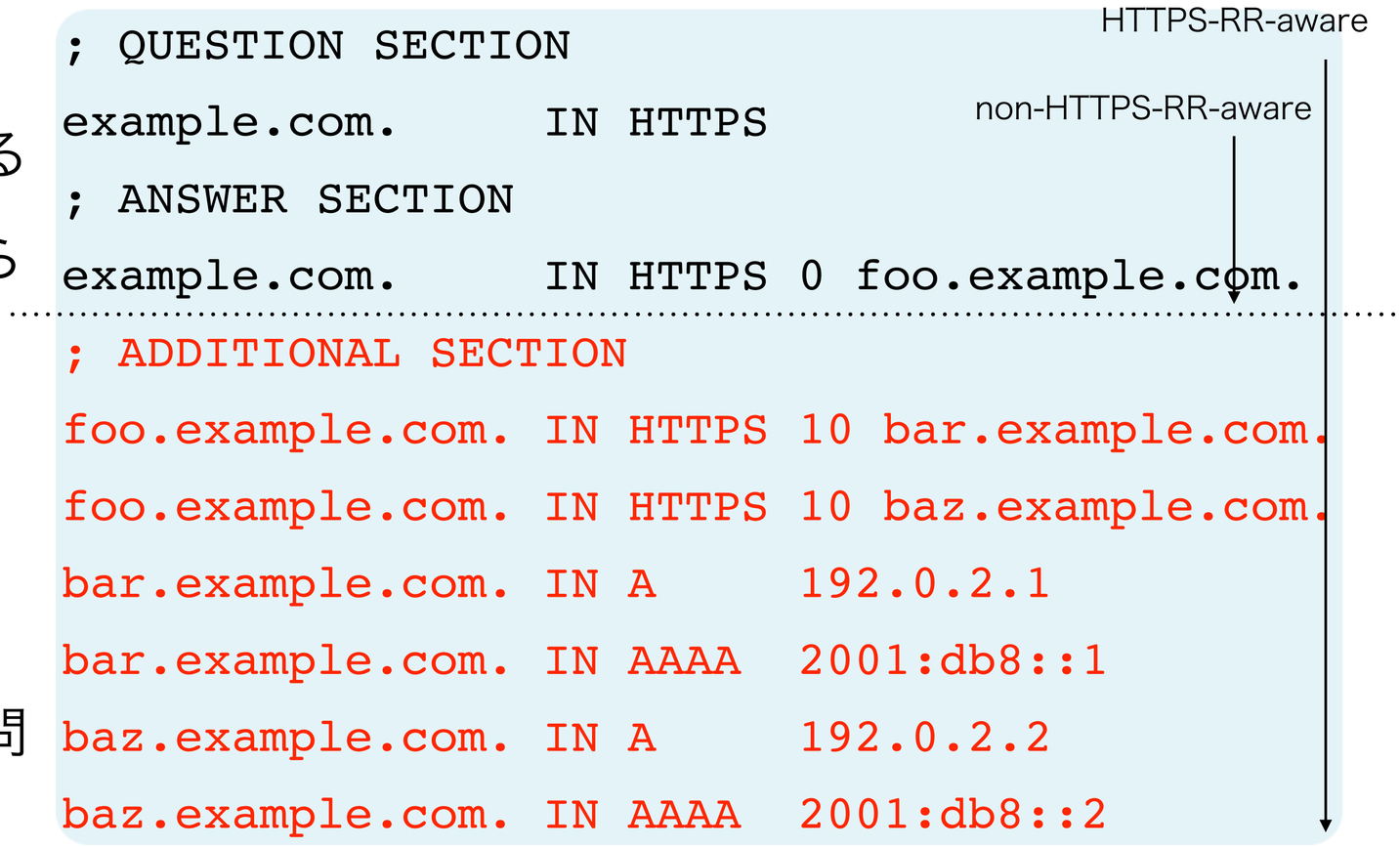
- クライアント側でも名前解決結果をキャッシュする(SHOULD)
- HTTPS RRと同時にそれっぽいA/AAAAも投機的に問い合わせ(SHOULD)
  - 「それっぽい」が正解なら名前解決にかかる時間を短縮できる
  - ハズレなら時間短縮にはならないし無駄になるけど
- HTTPS RR自体から得られるIPアドレスにより、A/AAAAの問い合わせを省略してもよい(MAY)

```
example.com. IN HTTPS 1 . ipv4hint=192.0.2.1 ipv6hint=2001:db8::1
```

  - A/AAAAの登録を省略できるわけではない

ADDITIONAL SECTIONの利用

- 現状
  - サーバは聞かれたことだけ答える
  - クライアントはIPアドレスが得られるまで問い合わせを繰り返す
- 将来
  - サーバは聞かれていない情報をadditional sectionに載せる
  - クライアントはこれを利用して問い合わせを減らすことができる



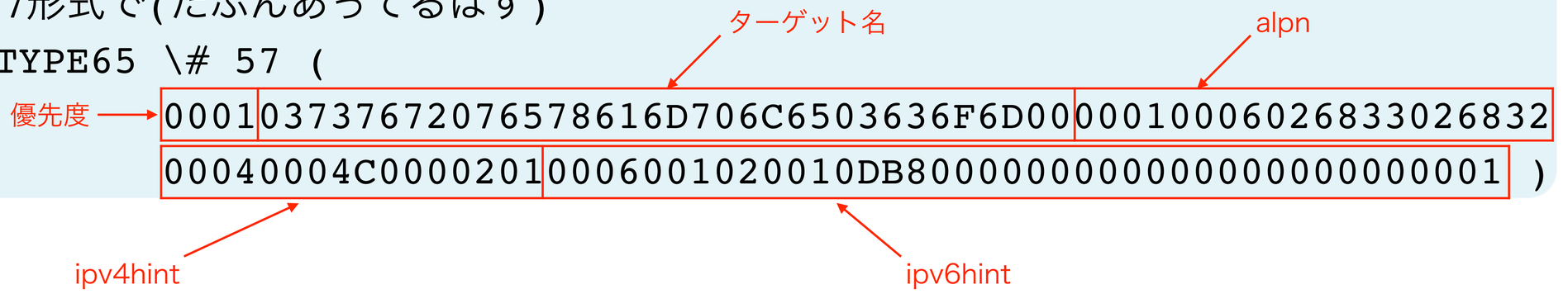
実際にDNSに登録したい!

- HTTPS RRの問い合わせに対して、サーバに特別な応答処理は不要
  - ADDITIONAL SECTIONに追加情報を載せられればうれしいけど、なくても問題なし
- 権威サーバ(セカンダリ)、フルリゾルバは既存のものがそのまま使える
  - ただし、KnotはセカンダリでのHTTPS RRの扱いにバグがあった模様(修正済み)
- 権威サーバ(プライマリ)のゾーン登録部分は要対応
  - ゾーンファイルに書かれたHTTPS RRのパース、文法チェック
  - PowerDNS4.4.0が対応済み
  - BIND、NSD、Knotは年内には対応版が出るんじゃないかなぁ

未対応権威サーバでの利用

- RFC3597形式でゾーンに書けば、未対応実装でもちゃんと使える
- が、まあ、やめとけ

```
; HTTPS RR
example.com. IN HTTPS  1 svr.example.com. (
                         alpn=h3,h2 ipv4hint=192.0.2.1 ipv6hint=2001:db8::1 )
; 同じものをRFC3597形式で(たぶんあってるはず)
example.com. IN TYPE65 \# 57 (
                         0001 03737672076578616D706C6503636F6D00 0001000602683302683 2
                         00040004C0000201 0006001020010DB800000000000000000000001 )
```



過渡期の利用

- 普及までの過渡期は、HTTPS RR対応/未対応クライアントが混在する
- どちらでも使えるような配慮が必要
  - CNAMEを使える場合はエイリアスモードを使わずにCNAMEで
  - ゾーン頂点でのエイリアスモードは無理してでもHTTPSとA/AAAAの併記で

```
example.com.      IN HTTPS 0 svr.example.com. ; HTTPS RR対応クライアントには別名を
                  IN A     192.0.2.1          ; 未対応クライアントにはIPアドレスを
                  IN AAAA  2001:db8::1
```

  - サービスモードはサービスパラメータの記載に利用目的を絞る

```
svr.example.com. IN HTTPS 1 . alpn=h3,h2     ; ターゲット名を “.” にすることで、
                 IN A     192.0.2.1          ; HTTPS RRの対応有無にかかわらず
                 IN AAAA  2001:db8::1        ; 同じA/AAAAを参照させる
```

HTTPS RRに対応してください

- まだドラフトですが、すでにたくさん使われています

- ブラウザでの対応は進んでいますが、それ以外でも対応は必要です

HTTPクライアント

- Webブラウザ
  - Safariは対応済みでクエリ出しまくってる
  - 対応済みだがまだデフォルト無効のChromeやFirefoxも、早い時期にデフォ有効に変更してくると思われる
- その他HTTPアクセスをする各種アプリケーション、ライブラリ
  - 従来のA/AAAAにしか対応していないものは、ブラウザでのアクセスと異なるサーバに接続して意図したHTTPレスポンスを受けとれない可能性が出てくる

ファイアウォール

- 未知のDNSレコードタイプは怪しいからと問い合わせパケットを捨ててしまうファイアウォールで、HTTPS RRを巻き添えにしている事例がある模様
- 中間者攻撃として認識されると、A/AAAAの名前解決ができたとしてもダウングレードを避けるためにアクセスしなくなる
- 中間者攻撃として認識されなくても、A/AAAAが使われるのはドロップされたHTTPS RRの名前解決がタイムアウトしてから
- SVCB/HTTPS RR (Type64/65)はドロップせずに通過させるようファイアウォールの設定を変更してください

DNSサーバ、DNSフォワーダ

- A/AAAAが存在し、かつHTTPS RRが存在しない名前のHTTPS RR(Type65)を問い合わせたときに、NOERRORを返すことを確認
  - 確認コマンド → `dig @サーバ type65 {A/AAAAだけ存在する名前}`
  - HTTPS RRにNXDOMAINを返すと、A/AAAAも存在しないと判断されてしまう
  - HTTPS RRを捨てるファイアウォールが存在しないことも確認する
- 将来的にはadditional sectionに情報を載せられるような実装への入れ替え(バージョンアップ)を検討

権威DNSサーバ

- 普及するまでの過渡期は、HTTPS RR対応/非対応ブラウザどちらでもアクセスできるような配慮が必要
  - ぶっちゃけ、HTTPS RRを使わずこれまでどおりA/AAAAだけ使っていれば問題なし
  - もちろんその場合はHTTPS RRのメリットを享受できない
- 将来、CDN/xSP事業者がユーザに対してCNAMEではなくHTTPS RRで登録するよう求めてくると思われる
  - それに対応できるよう、DNSホスティングサービス事業者はHTTPS RRを登録できるようにしておく必要がある
  - 自前運用であれば、自分が必要になってからでよい

まとめ

- WebサーバのDNSへの登録方法が変わるよ
- SVCBとHTTPSというリソースレコードがもうすぐRFCに
  - ゾーン頂点でも使える別名
  - MXやSRVのような優先度指定
  - HTTPバージョンやECH鍵などのサーバ構成情報の記載
  - などの機能
- とくにファイアウォールでドロップしてないかは今すぐ確認を