IPv6逆引き自動生成 DNSサーバ

藤原和典
株式会社日本レジストリサービス(JPRS)
fujiwara@jprs.co.jp

IPアドレスの逆引き

- IPアドレスからホスト名への逆引き
  IPアドレスに対応するドメイン名に
  PTRリソースレコードでホスト名記述

- IPv4: 192.168.1.3の逆引き
  3.1.168.192.in-addr.arpa IN PTR host.example.jp.

- IPv6: 2001:db8:21::123:4567:89ab:cdefの逆引き
  f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.0.0.0.0.0.1.2.0.0.8.b.d.0.1.0.0.2.ip6.arpa IN PTR host.example.jp.

IPv6アドレスの逆引きドメイン名

入力はIPv6アドレス

2001:db8:21::123:4567:89ab:cdef　　(IPv6アドレス)

手順1 ::による省略をやめ、0を省略しているところを補う

2001:0db8:0021:0000:0123:4567:89ab:cdef (32文字+ ：)

手順2 :を削除し、各文字の間に.を追加

2.0.0.1.0.d.b.8.0.0.2.1.0.0.0.0.0.0.1.2.3.4.5.6.7.8.9.a.b.c.d.e.f

手順3 逆順に並べ替え

f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.0.0.0.0.0.0.1.2.0.0.8.b.d.0.1.0.0.2

手順4 末尾に.ip6.arpaを追加したものがIPv6での逆引きドメイン名

f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.0.0.0.0.0.0.1.2.0.0.8.b.d.0.1.0.0.2.ip6.arpa

IPアドレスの逆引きの利用シーン

- ログ解析など
  - どこのドメイン名から来ているか調べるのが容易
- 運用支援
  - who, last コマンドの出力
  - Tracerouteの出力
- アクセス制限、認証
  - ある組織からだけ見せるページをつくりたい
    - Allow from *.example.co.jp
  - rlogin/rsh/RPOPなど
  - SMTPでの接続元確認
    - 逆引きが変なマシンからは送れない
      - (逆引きは別料金のため設定してない)

逆正一致 (PARANOID check)

- DNS逆引きで得たホスト名からさらにIPアドレスを検索し、もとのIPアドレスが含まれているかを確認する手法
  - 逆引きには、逆引きドメイン名の管理者が任意のホスト名を記述できるため、ホスト名だけで信用すると危険
    - ホスト名だけで認証するようなサービス = RPOP, rsh, rloginなど
    - 大昔は逆引きをなりすまして.rhostsの記述を推定してloginできた
  - 正引きにIPアドレスが書かれていると、逆引きと正引き両方の管理権限があると考えることができる

IPv4での逆引き設定

- 多くのISPでは、ユーザに割り当てるすべてのアドレスにISPのドメイン名がついた逆引きを設定している
- 逆引き結果のドメイン名にIPアドレスを対応づけ、逆正一致チェックも通る

- IPv6でも、ユーザに割り当てたアドレス空間の逆引きは必要か？
- その場合、逆正一致チェックも行うか？
  - 従来のIPv6対応ではv4と同じチェックを行っている

IPv6での逆引きの特徴

- IPv6では、利用者に割り当てられるアドレスが膨大
  - すべてのアドレスに逆引きを設定することは困難
    - /64のアドレス数は2^64 = 約1.8*10^19 = 約1800京 = 約18E(エクサ)
    - ゾーンファイルに書くと、一行100バイトで1800Eバイト
- 利用者の使用するアドレスが短時間に変化
  - Privacy Extension for Stateless Address Autoconfiguration in IPv6 (RFC 4941)
  - Windows Vistaなどでは実装済み

IPv6での逆引き設定の考え方

案1 すべてのアドレスの逆引きを事前設定
- /64ですら18エクサバイト必要なので不可能

案2 使用されるアドレスだけ動的に追加設定
- Neighbor Discoveryを見張って追加
- Windows XP/VistaはDynamic Updateを送るので受け付ければよい

案3 利用者に権限委任

案4 /64などの単位でwildcardにPTR設定
- 逆正一致チェックは通らない

案5 IPv6アドレスの逆引きと対応する正引きを自動生成する専用のDNSサーバを製作

draft-howard-isp-ip6rdns-00

- Reverse DNS in IPv6 for Internet Service Providers
- Time Warner CableのL. Howard氏とComcastのA. Durandから2009/6/9にIETFに提出
- 概要
  - IPv4での逆引き設定
  - IPv6での逆引き設定の考え方
    - 前ページの案1から案5
  - IPv6では利用者のアドレスの逆引きをISPが提供することをやめることを推奨する提案
  - 動的に逆引きを生成するのがよくない理由はDNSSECのため
- IETF75での議論は発散
  - 現在dnsop mailing listで議論再開

逆引き自動生成:要件

- すべてのIPv6アドレスに、ユニークなホスト名を与え、その正引きも用意すること
  - /32～/64ぐらいの範囲の逆引きを生成できること
  - 対応する正引きも生成すること
  - IPv6アドレス、逆引き、正引きのホスト名は一対一対応すること
  - DNSサーバとしてRFCに準拠した応答を生成すること

逆引き自動生成: 設計

- IPv6アドレスに対応するホスト名ラベルを定義
  - 32文字の十六進文字列
  - IPv6アドレスの16進表記から、0の省略をやめ、: を削除したもの
    - IPv6アドレスと一対一対応
- 例
  - IPv6アドレス 2001:db8:1234::123:4567:89ab:cdef
  - ドメイン名を user.example.jp とする
  - ホスト名ラベルは 20010db81234000000123456789abcdef
  - ホスト名は、20010db81234000000123456789abcdef.user.example.jp
  - DNSには以下のRRが記述されることとなる
    f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.0.0.0.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa IN PTR 20010db81234000000123456789abcdef.user.example.jp
    20010db81234000000123456789abcdef.user.example.jp IN AAAA 2001:db8:1234::123:4567:89ab:cdef

逆引き自動生成: 試作

- 試作なのでPerl　Net::DNS::Nameserver使用
  - サーバの準備、パケットの分解、組み立て機能あり
  - DNS queryから、応答パケットの中身をつくるReplyHandlerを作成するだけ
- 最初のバージョンは1時間で完成
  - コメントや空行を除いて104行程度
- 今回の試作の制約 (=手抜き)
  - 対応ゾーンはひとつのみ →複数ゾーンなら複数サーバで
  - DNSサーバは外部名のみ
  - 応答性能が悪い (Perlだから) → Cで書けば性能20倍

逆引き自動生成: 応答の設計 (1)

- 例: 2001:db8:1234::/48, user.example.jp

- 指定された逆引きゾーンのNS, SOAを応答
  - 4.3.2.1.8.d.b.0.1.0.0.2.in-addr.arpa
    - SOA v6rev.example.jp. Postmaster.v6rev.example.jp. 1 3600 900 86400 900
    - NS v6rev.example.jp

- 指定された範囲の逆引きのPTRを応答
  f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.0.0.0.0.0.4.3.2.1.8.b.d.0.1.0.0.2.ip6.arpa IN PTR 20010db8123400000123456789abcdef.user.example.jp

逆引き自動生成: 応答の設計 (2)

- 指定された正引きゾーンのNS, SOAを応答
  - user.example.jp
    - SOA v6rev.example.jp. postmaster.v6rev.example.jp. 1 3600 900 86400 900
    - NS v6rev.example.jp
- 指定された正引きゾーンのAAAAを応答
  - 20010db81234000000123456789abcdef.user.example.jp IN AAAA 2001:0db8:1234:0000:0123:4567:89ab:cdef
  - ラベルがIPv6アドレスそのものなので、そのままAAAAとして応答

逆引き自動生成: 結果1

- 実際に使用した
  - 2001:200:132:6::/64 の逆引きを user.dnslab.jp 以下のホスト名として設定
  - JPRSの研究ネットワークのーセグメント
  - Perl scriptをjail環境下で、v6rev.dnslab.jpのホスト名で動作させた

逆引き自動生成: 結果2

- dig結果あれこれ

```
# dig @v6rev.dnslab.jp +short -x 2001:200:132:6::1
2001020001320006000000000000001.user.dnslab.jp.
# dig +short @v6rev.dnslab.jp
    2001020001320006000000000000001.user.dnslab.
jp. aaaa
2001:200:132:6::1
# dig +short @v6rev.dnslab.jp 6.0.0.0.2.3.1.0.0.0.2.0.1.0.0.2.ip6.arpa ns
v6rev.dnslab.jp.
# dig +short @v6rev.dnslab.jp 6.0.0.0.2.3.1.0.0.0.2.0.1.0.0.2.ip6.arpa soa
v6rev.dnslab.jp. postmaster.v6rev.dnslab.jp. 1 3600 900 86400 900
# dig +short @v6rev.dnslab.jp user.dnslab.jp ns
v6rev.dnslab.jp.
# dig +short @v6rev.dnslab.jp user.dnslab.jp. soa
v6rev.dnslab.jp. postmaster.v6rev.dnslab.jp. 1 3600 900 86400 900
```

逆引き自動生成: 結果3

- あるホストからsh.wide.ad.jpへloginしたところ

  sh% who

  fujiwara ttypb Sep 1 14:37 (200102000132000600000000000006.user.dnslab.jp)

  sh% last fujiwara | head -1

  fujiwara ttypb 200102000132000600 Tue Sep 01 14:37 still logged in

- queryperfによる応答性能
  - 557 queries / sec
  - 存在する逆引きと、正引きAAAA検索

逆引き自動生成: 使い方

- DNSサーバの専用IPアドレス、ホスト名準備
  - 中でbindさせるんで、独立させる必要はないが、jailとかchroot、vmなどで分離させるのが安全
- Perl 5.8 + Net::DNSの導入
- v6rev.plをカスタマイズ
- Rootで、v6rev.plを起動

v6rev.pl: カスタマイズ

```
my $nsname = ‘ns.example.jp’;              # DNSサーバ名
my $mydom = ‘user.example.jp’;             # 正引きゾーン名
my $myrev = ‘8.b.d.0.1.0.0.2.ip6.arpa’;  # 逆引きゾーン名
my $ns = Net::DNS::Nameserver->new(
    LocalAddr     => [ ‘127.0.0.1’, ‘::1’ ],  # DNSサーバアドレス
    LocalPort     => 53,                      #          ポート番号
    ReplyHandler  => ¥&reply_handler,
    Verbose       => 0,
);
```

注: 複数のDNSサーバを用意するときはその他の修正が必要

まとめ

- ユーザに割り当てたIPv6アドレスの逆引きとして、IPv4のときのような機械的なホスト名を自動生成することは可能である

- 応答性能については、Cで実装すれば十分な性能が得られる

質問

- このような機械的な逆引きホスト名は必要でしょうか？

  20010200013200060123456789abcdef.user.dnslab.jp

- 今後、なんらかの対応が必要でしょうか？
  - だれか C で書いてくれるひとはいらっしゃいますか？
  - IETFのdnsop WGで、逆引き不要論と戦う？

- ISCの権威DNSサーバへくる逆引きクエリはどれぐらいでしょうか？ (query/sec)