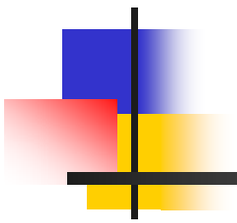
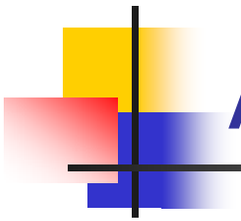


BSD Unixにおいて IPv6 を有効にした際に発生する課題とその対策

WIDE Project /アラクサラネットワークス(株)
鈴木伸介 <suz@kame.net>



Abstract

- DNS
  - AAAA Queryに対する異常応答
  - アドレスファミリー未指定時のA/AAAA Query順序
  - DNSサーバアドレス検出
- まとめ

AAAA Queryに対する異常応答

- AAAA Queryにより異常な応答をするDNSサーバがあると、AAAA Queryを行ったときのタイムアウト待ちにより、通信遅延が発生 (RFC4074)

IPv6で問い合わせたとき限定の症状	*BSDでの対応
Queryが無視される	A/AAAA Queryの順番を細工 (抜本解ではないが…)
エラーメッセージが返ってくる (ホスト名不在=NXDOMAIN)	
Lame Delegationになる	
エラーメッセージが返ってくる (「ホスト名不在」以外)	エラーメッセージをトリガーに次のアクション (アプリケーション依存)
壊れたrecordが返ってくる	壊れたrecordを廃棄

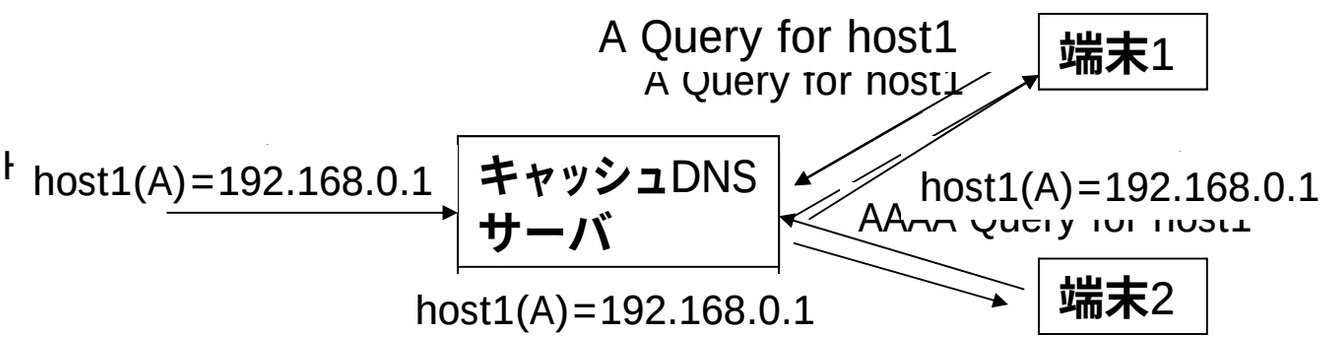
アドレスファミリー未指定時の A/AAAA Query順序の工夫

- 一般的にはアプリケーション依存
  - 普通は*BSD付属のライブラリの実装依存

OS	Query順序
NetBSD, OpenBSD, FreeBSD (~5.3)	AAAA→A
FreeBSD (5.4~)	A→AAAA
KAME SNAP	非link-local IPv6アドレスの有無で順序を変える あり: AAAA→A なし: A→AAAA

A/AAAA Query順序の調整だけでは回避しきれないケース

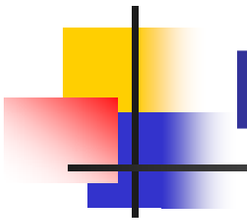
- **誰かが**AAAA Query**する限り、「ホスト名不在」エラーには対応不能**
  - **端末が**A Query, AAAA Query**を送信**
  - A Query**により、**IPv4**アドレスを学習し通信** (OK)
  - AAAA Query**により、キャッシュ**DNS**サーバに「ホスト名不在」による** negative cache**生成**
  - **以降キャッシュ**DNS**サーバに**A Query**しても、**IPv4**アドレスを学習できない** (NG)



A/AAAA Query順序の調整だけでは回避しきれないケース (cont.)

- AAAA Queryを行う限り「答えのないAAAA Queryのタイムアウト待ち」は避けられない
  - KAME SNAPでは、A Queryの応答時間から適当なタイムアウト値を推測し、タイムアウト時間を必要最小限にしている。

- いずれのケースも端末側ではどうしようもない
  - AAAA Queryに異常応答をするDNSサーバの修正が必須
  - 駄目な場合は、アプリケーション毎の対応が不可避 (e.g. mozilla)

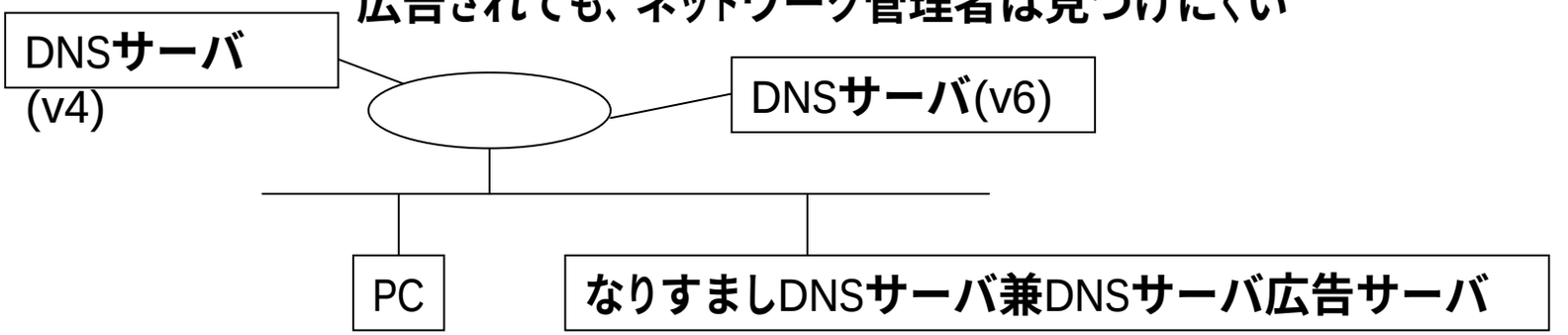


DNSサーバアドレス検出

- **各配布方法への対応状況**
  - RA**配布**: **未対応**
  - DHCPv6**配布**: **対応済** (WIDE-DHCPv6)
  - Well-known Anycast address: **対応済 (手設定)**
- **根本的な問題**
  - **どれが標準**?
    - IETF**でも標準化作業が頓挫**…
  - **仮に標準が決まったとして**
    - DNS**サーバの**IPv4**アドレス**,IPv6**アドレスがあるときどちらを優先すべきか**?

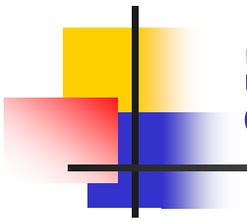
DNSサーバのIPv4/IPv6アドレスの優先度問題

- IPv4/v6 Dual-Stack化により顕在化
  - DNSサーバアドレスをDHCPv4,v6両方で学習
  - c.f. 類似問題
    - DNSサーチパスをDHCPv4,v6両方で学習
    - Policy Tableを複数の上流ISPから学習
    - Default Routerを複数の隣接ルータから学習
- 想定される問題
  - Queryの回数が無駄に増える
  - 「なりすましDNSサーバ」への誘導に悪用することも可能
    - 特にIPv4 onlyセグメント内で「なりすましDNSサーバのIPv6アドレス」が広告されても、ネットワーク管理者は見つけにくい



*BSDでのDNSサーバのIPv4/IPv6アドレスの優先度

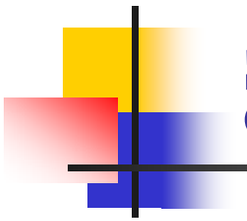
- 通常はIPv4アドレスだけ使用
  - OS付属のDHCPv4 clientでDNS情報取得
  - DHCPv6クライアント(WIDE-DHCPv6)も、デフォルトでは取得したDNS情報を端末に反映しない
    - 必要な人だけ、DHCPv6 clientで取得した情報を/etc/resolv.conf へ反映するよう設定
- 実用上はこれで特に問題ないでしょう



まとめ

- DNS関連の問題は以下の2つにより対処
  - 壊れたメッセージを廃棄
  - A/AAAA Queryの順序を工夫している
    - が端末側の対応には限界がある→サーバ側の対応を切に望みます
- DNSサーバアドレス検出
  - 実用上はDHCPv4のみで十分

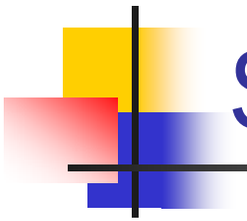
Thanks!



まとめ

- DNS関連の問題は以下の2つにより対処
  - 壊れたメッセージを廃棄
  - A/AAAA Queryの順序を工夫している
    - が端末側の対応には限界がある→サーバ側の対応を切に望みます
- DNSサーバアドレス検出
  - 実用上はDHCPv4のみで十分
- Source Address Selection
  - 一部実装済
  - 動的Source Address Selection Policyは、複雑な割に有効な局面が少ない感がある
- Default Gateway Selection
  - Router-Preferenceは実装済
  - More-Specific Route optionは未実装だが、受信のみRIPngで十分

SWG指摘の他のネタに対するコメント

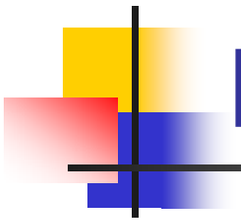


Source Address Selection

- RFC3484**実装状況**
  - longest-match rule = **全ての***BSD**で実装済**
  - Policy Table = **一部の***BSD**で実装済**
    - FreeBSD: 5.2~
    - NetBSD: **まだ** (KAME-SNAP**では実装済**)
    - OpenBSD: **まだ** (KAME-SNAP**では実装済**)
    - **ただしいずれも手動設定** (ip6addrctl)**で、**DHCPv6**などと連動した自動設定は未サポート**

Source Address Selection (cont.)

- Policy Table**自動設定が未サポートな背景**
  - **標準化がまだ**
    - **汎用的な**Policy Table**自動設定は非常に難しい**
      - IPv6**マルチホーム問題そのもの**
      - **一部の場合**(e.g. **閉域網と**Internet**の同時使用**)**には簡単かつ効果的**
    - **上の効果的なパターンは「**Unique Local Address (RFC4193) **と**longest-match rule**の併用」でも対応可**
      - Policy Table**自動設定ならば**/48**よりも広いアドレス空間でも有効**
      - **そのメリットも、**FC00::/8 (centrally-managed Unique Local Address) **の**Registry**割当が始まればなくなる可能性大**



Default Gateway Selection

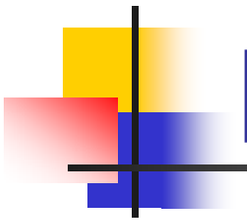
- Router Preference
  - **ルータ側 = 全*BSD対応済**
  - **端末側 = 一部BSD端末で使用可能**
    - FreeBSD: 6.1**～**
    - NetBSD: -current (Jan 2006)
    - OpenBSD: (KAME-SNAP**のみ**)
- More-Specific Route
  - **ルータ側 = 全*BSD対応済**
  - **端末側 = 未実装**

Default Gateway Selection (cont.)

- **なぜ端末側で**More Specific Route**が未実装か**?
  - kernel**内実装が困難**
    - **経路制御プロトコルを**kernel**内で実装するのとほぼ等価**
  - **端末で「受信のみ**RIPng**」を動かすほうが、素直かつきめ細かい制御ができるのでは**?
    - **素直=これなら、***BSD**全て対応済み**
    - **きめ細かい制御=経路制御計算を踏まえた経路広告**

到達性が無いIPv6アドレス取得

- 基本的にはアプリケーション依存
  - OSはアプリケーションにエラーを返す
    - host unreachable, net unreachable, ...
  - あとはアプリケーションがそのエラー値を見て賢く振舞うか次第
- アプリケーションではエラー処理はちゃんと実装しましょう
  - IPv4でも同じことは起こる

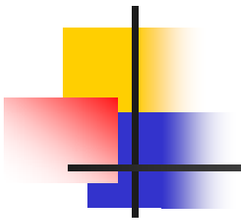


自動トンネル

- ***BSDではユーザが意図的に有効にしない限り、設定されない**
  - 6to4
  - ISATAP (KAME)
  - TSP (freenet6)
  - L2TP (l2tpd)
- **そのため、指摘されたような問題は発生しない**

マルチキャストとPersonal Firewall

- 現状
  - 「マルチキャストだったらデフォルト廃棄」という Personal Firewall実装もあるが、これは非常に迷惑
- 提案
  - こんなStateful inspectionが欲しい
    - MLD joinしたら、そのグループ宛のパケットだけ通す
    - その他のグループ宛のは廃棄



IPsecとmulticast

- 事前共有鍵による鍵交換は動く
- 動的鍵生成による鍵交換は未対応
  - IPv4/v6にかかわらない問題