ホスティングサービスのIPv6対応

Internet Week 2008@**秋葉原**
2008/11/26

(株)クララオンライン
白畑　真 <shin@clara.ad.jp>

今日のアジェンダ

- はじめに
- IPv6自体への対応
  - ネットワーク
  - アドレス設計
  - DNS
- 二つのデュアルスタック化
- サーバ単独のデュアルスタック構成
  - IPv4射影アドレス
  - 障害発生時の切り分け
- IPv4/IP6デュアルスタックサービスの検討
  - End-to-Endモデル
  - Middleboxモデル
  - Hybridモデル
- まとめ

今日の結論

- まずはIPv6の試験環境を
  - ネットワークとDNSのIPv6対応が必須
- サーバのデュアルスタック化と
  サービスのデュアルスタック化は別
  - サーバでのIPv4/IPv6デュアルスタックは
    イバラの道
  - サーバはシングルスタック運用、サービスは
    デュアルスタックがお勧め

はじめに
今日の議論の範囲とその前提

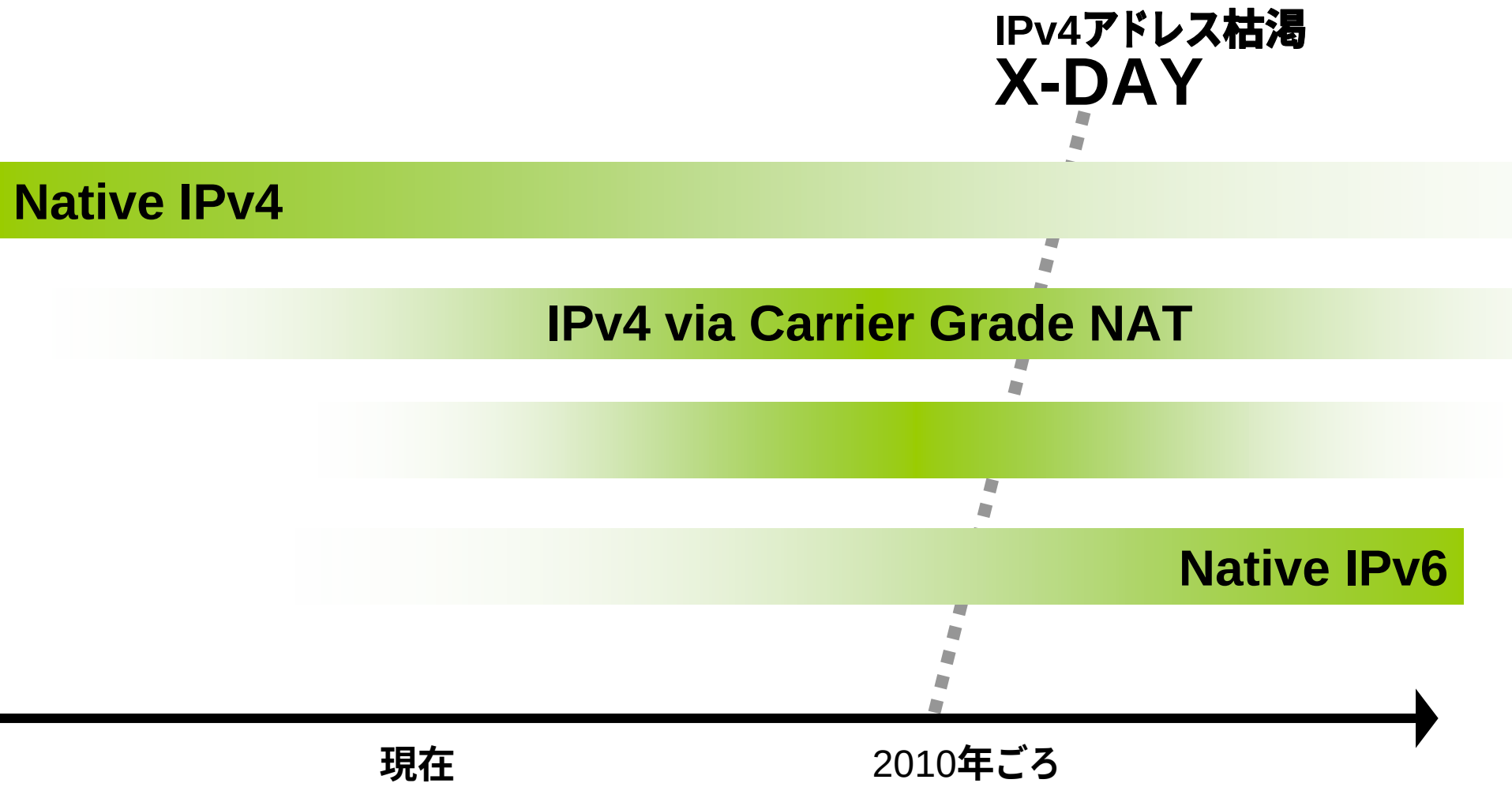
- IPv4/IPv6共存環境
  - IPv6クライアントのみを想定したサービスではなく、既存サービスのIPv6クライアント対応という視点で議論を進めます。
  - ユーザ側にとっては、アプリケーションやコンテンツが重要であり、IPv4/IPv6にどちらでもよいという前提に立ちます。
- 主としてレイヤ4～レイヤ7が対象
  - レイヤ3(インターネット接続)については、iDC側の構成に依存する部分ため、原則として所与の条件とします。
  - ただし、内部ネットワークについては必要に応じて議論します。

ご注意:
- 2008年11月現在、(株)クララオンラインはIPv6対応のサービスの提供は行っておりません。
- 本発表はIPv4/IP6デュアルスタックサービスの展開手法についての一検討であり、特定のプロダクトまたはサービスを推奨するものではありません。

ホスティングサービスの位置づけ

- ホスティングサービス
  - 定義:
    - 物理サーバもしくは仮想サーバ全て、もしくはそれらの資源の一部を提供するサービス
    - 例: Webサーバや電子メールサーバ
  - 標準化されたサービスを複数の顧客に対して提供
- iDC事業者との違い
  - ネットワーク接続性・ハードウェア・OS・ソフトウェアを含めてサービスとして提供
  - ホスティング事業者はiDCのユーザという位置づけ
    - 一部重複する場合もある

IPv4 アドレス枯渇問題とクライアント環境
(発表者による予測)



IPv4アドレス枯渇とホスティング事業者

考えられる対応策

1. IPv4アドレスの利用効率向上、移転自由化
   - 現在議論中。詳しくは夕方のB3セッションにて
2. ~~Carrier Grade NAT~~
   - iDC/ホスティング事業者には解決策とならない
3. IPv6対応

   IPv4/IPv6デュアルスタック対応
   - IPv4からのアクセスが(ほぼ)なくなるまでは、IPv4クライアントとIPv6クライアントの両方をサポートする必要がある

ホスティングサービスとIPv6
よく聞く声

- クライアント側がIPv6に対応していない
  - ISP側がIPv6に対応していない
    - IPv6ネットワークの品質がIPv4より低い
  - IPv6でサービスを提供してもアクセスされない
- iDCがIPv6サービスを提供してくれない
  - トンネル経由でIPv6を利用する方法もある
- ノウハウがない/何から初めていいかわからない
- 商用のツールがIPv6に対応していない
  - コントロールパネル、アクセスログ解析ソフトなど

IPv6の普及に応じた時期毎の対応

1. テスト導入期
   - 想定環境
     - 実験環境や試験サービス(アプリケーションやサービスのIPv6対応検証など)
   - 性能
     - トラフィック全体のうち、ごく一部がIPv6トラフィック
     - IPv6がとりあえず動作することが目的。性能については考慮しない
   - コスト: 低
2. 本格導入期
   - 想定環境
     - プロダクションレベルのIPv4/IPv6デュアルスタックサービス
     - IPv6シングルスタックでのサービス提供
   - 性能
     - トラフィック全体のうち、一定割合以上がIPv6トラフィック
     - 充分な検討が必要
   - コスト: 中〜大

検討事項

- 主に定性的な課題を検討
  - 事業形態ごとの課題の洗い出し
- IPv6対応時の主要な課題
  - ネットワークのIPv6対応
  - DNSのIPv6対応
  - ソフトウェアのIPv6対応
    - OSのIPv6対応
    - サーバアプリケーションのIPv6対応
    - Webアプリケーション・顧客システムのIPv6対応
  - バックエンドシステムのIPv6対応
- IPv6対応方式の検討
  - 時期毎に最適解が異なる

CLARA ONLINE

IPv6 自体への対応

ネットワークのIPv6対応(1/2)
主な接続形態

- ネイティブ接続
  - iDC側の対応が必要
    - 本格的導入期向け
    - 通信品質を保証しやすい
  - シングルスタック接続
    - IPv6ネットワークのみの接続性
  - デュアルスタック接続
    - IPv4/IPv6ネットワーク両方の接続性
- トンネル接続
  - テスト導入期向け
  - 導入のハードルが低い
    - iDC側のIPv6対応は必ずしも必須ではない
      IPv6ネイティブ接続が利用できない場合でも、IPv4ネットワーク経由でIPv6ネットワークへの接続性を確保できる
  - 通信品質の保証が難しい

ネットワークのIPv6対応(2/2)
トンネリングの手法

- 主なトンネリングの接続形態
  - 静的トンネル
    - iDC/キャリアのトンネリングサービス
    - IPv6トンネルブローカー
  - 動的トンネル
    - 6to4
      - IPv4グローバルアドレスを埋め込んだIPv6アドレスを利用し、自動的にIPv6 over IPv4トンネルを構築
      - BGP Anycastで世界中の6to4リレールータがトラフィックを変換
      - どのルータを経由するかわからないため、あくまで実験向け
    - 他の動的トンネル技術としてはISATAPやTeredoがある
      - IPv6未対応のiDCにあるサーバで利用する目的には向かない
  - トンネルと通信品質
    - 海外のルータを経由してトンネリングを行った場合、国内で通信が終端する場合と比較して通信品質が低下
    - RTT、パケットロス、安定性

IPv6アドレッシング(1/2)
アドレスの設定方法

- **手動設定**
- **自動構成**
  - EUI-64**により下位**64bit**のインタフェース**ID**を自動生成**
    - NIC**の交換時に**IPv6**アドレスが変更になる**
    - OUI(Organizationally Unique Identifier)**部分から機器のメーカーを知ることができる**
  - **匿名アドレス/一時アドレス**
    - IPv6**アドレスを一定時間で使い捨て**
    - RFC3041: Privacy Extensions for Stateless Address Autoconfiguration in IPv6
- **サーバでは、RAと匿名アドレスによるIPv6アドレスの構成は無効化した方がよい**

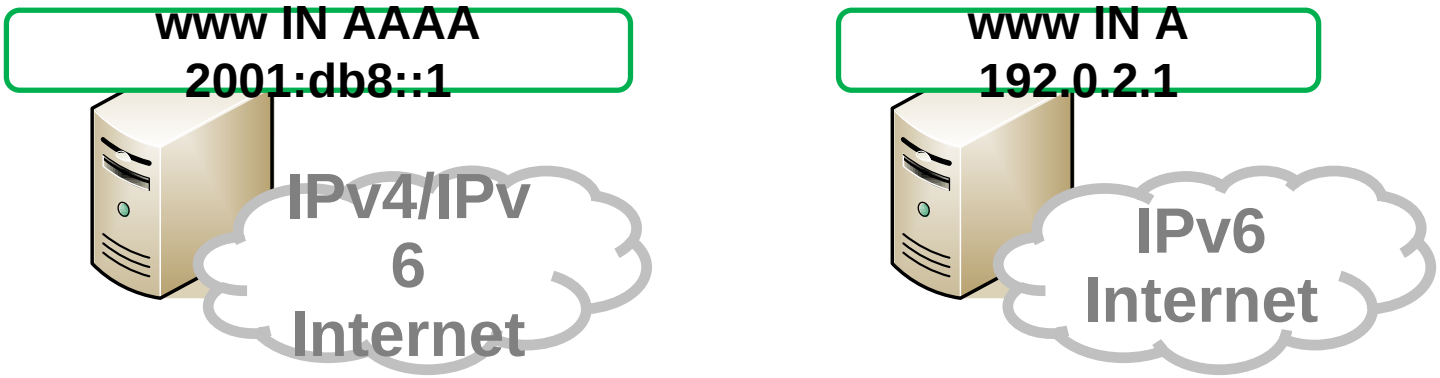
IPv6アドレッシング(2/2)
アドレスの設計と活用

- **サービスに紐づけたアドレッシング**
  - **広大なアドレス空間を利用し、アドレス自体に意味を持たせる設計が可能**
    - **例**:
      - Web**サーバ** 2001:db8::80
      - DNS**サーバ** 2001:db8::53
- **各顧客に割り当てるアドレスサイズの検討**
  - /128
    - IPv4**でのサービスと同等**
  - /64
    - **サブネット単位の割り当て**
      IPv6**ならでは広大なアドレス空間を利用できる**
      ネットワーク設計の変更が必要
- **二種類のアドレス**
  - **Global アドレス**
  - Link-local**アドレス**
    - **同一リンク上でのみ利用可能**

DNSのIPv6対応(1/5)

IPv6環境でのDNS

- **正引き**: **ホスト名から**IPv6**アドレスへの変換**
  - AAAA**レコードで**IPv6**アドレスを記述**
- **逆引き**: IPv6**アドレスからホスト名への変換**
  - ip6.arpa**以下のツリーに**PTR**レコードを記述**



DNSレコードとDNSトランスポートは独立

1. IPv4/IPv6**のいずれかを問わず、DNSはIPv6アドレスの問い合わせ(AAAAレコード)に応答する**
2. **IPv6の通信で、**DNS**が**IPv4(A**レコード)と**IPv6(AAAA**レコード) の問い合わせに応答する**

DNSのIPv6対応(2/5)

DNSのIPv6対応に必要な機能

A) リソースレコードのIPv6対応
- AAAAレコードが記述できること
  - ASPサービスを利用している場合には事業者の対応が必要
- ip6.arpaレコードの逆引きが設定できること
- DNS権威サーバ自体への到達性はIPv4のみでも良い

B) EDNS0 (Extension Mechanisms for DNS) 対応
- もともとのDNSでは、UDPパケットのデータ長は最大512バイトであるため、512バイトより大きいデータ長に対応するための拡張
  - IPv6リソースレコードと直接関係はないが、IPv6アドレスなどサイズの大きなレコードを格納する際に役立つ
  - 互換性の問題があるため、Root DNS/ccTLD DNSでは避けられている

C) DNS権威サーバのトランスポートのIPv6対応
- IPv6インターネット経由でDNSクエリに応答

- 主要な実装:
  - 1のみ対応: djbdns
  - 1~3すべて対応: BIND 9, NSD, Microsoft DNS (Windows Server 2003)

DNSのIPv6対応(3/5)
名前の付け方の例

1. IPv4向けとIPv6向けで同じ名前を使う
   - 例: KAME Project
     - www.kame.net (Aレコードと AAAAレコードの併用)
       - 203.178.141.194
       - 2001:200:0:8002:203:47ff:fea5:3085
2. IPv4向けとIPv6向けで別の名前を使う
   - 例: Google
     - www.google.com (Aレコードのみ)
       - 72.14.205.147
       - 72.14.205.99
       - 72.14.205.103
       - 72.14.205.104
     - ipv6.google.com (AAAAレコードのみ)
       - 2001:4860:0:2001::68

DNSのIPv6対応(4/5)
名前の付け方

1. 同じ名前を使う

- メリット:
  - 透過性: ユーザがIPv4/IPv6を意識することなくサービスを利用できる
- デメリット:
  - 一部のユーザ環境から、名前解決に失敗する、もしくは遅くなる恐れ
    - DNS検索過程において壊れたDNSサーバが存在する
    - IPv4/IPv6で両方でサービスが提供されている場合、通信品質が低いプロトコル(現状ではIPv6)が選択される可能性がある
    - AレコードとAAAAレコードの問い合わせ順序によっては、遅延が発生する場合
    - IPv6の閉域網とIPv4インターネットの組み合わせ: マルチプレフィックス問題や、RAによりトンネルが利用できない問題の影響を受ける恐れ

2. 別の名前を使う

- メリット:
  - レコードの設定の柔軟性が増す
  - 壊れたDNSサーバの実装に伴う問題を回避できる
    - AAAAレコードの問い合わせに対する応答を正しく処理できないDNSサーバ
- デメリット:
  - 透過性: ユーザがIPv4/IPv6のいずれを利用するか意識する必要がある

iDC/ISP以外のDNSサーバにも注意

DNSサーバを内蔵しているロードバランサやブロードバンドルータ、DNSの内容を検査するFirewallやIDS,IPSについても配慮が必要

DNSのIPv6対応(5/5)
ドメイン名のレジストリ/レジストラの対応

※本ページではトランスポートとしてIPv6を利用して
再帰的問い合わせを行うケースを想定しています

- Root DNSにIPv6 Glueレコードが登録された
  - . (root) (2008年2月4日)
  - .jp / .kr (2004年7月20日)
- ドメイン名のレジストラの対応
  - IPv6のホスト登録ができるか
    - GlueレコードとしてAAAAレコードを登録できるか
  - JPRS (.jp), Verisign GRS(.com, .net) など主要レジストリは対応済み
- 対応レジストラー一覧:
  - FAQ : DNS : Which DNS Registrars allow me to add AAAA glue for my Domain Name Servers?
    http://www.sixxs.net/faq/dns/?faq=ipv6glue
  - network/IPv6/IPv6対応のレジストラー一覧 - Tomocha WikiPlus
    http://wiki.tomocha.net/ipv6_registrar.html

IPv6によるDNSサーバの運用

- RFC3901: DNS IPv6 Transport Operational Guidelines
  - *“Having DNS data available on both transports is the best situation.”*
    IPv4/IPv6の両トランスポートでDNSデータを持つことが望ましい
  - *“every recursive name server SHOULD be either IPv4-only or dual stack”*
    すべてのIPv4再帰ネームサーバはIPv4のみかデュアルスタックであるべき
  - *“every DNS zone SHOULD be served by at least one IPv4-reachable authoritative name server.”*
    すべてのDNSゾーンは、最低でも1つのIPv4到達性のある権威サーバによって提供されるべき
- RFC4472: Operational Considerations and Issues with IPv6 DNS
- **名前空間の分断を避ける**
  - **トランスポートがIPv4でもIPv6でも、同じリソースレコードを提供**
    - **IPv6からのみ到達性のあるネームサーバだけを運用した場合、IPv4インターネットからは参照できなくなる**
    - **必ずIPv4で到達性がある権威サーバを用意する**

アプリケーションのIPv6対応 (1/2)
アプリケーションそのもののIPv6対応

- IPv6対応
  - ソケットレベルでの変更
    - bindシステムコールで設定するソケットの変更(後述)
- 設計変更の検討
  - IPv6アドレスの表記にまつわる課題
    - ログ取得、アクセスログ解析、設定ファイル..
    - IPv6アドレスならではの課題
      - クライアント側でIPv6一時アドレス (匿名アドレス)を利用している場合、同一ホストのIPv6アドレスが変化する
    - 構成上の課題
      - プロトコルトランスレータやリバースプロキシを利用する構成では、アクセス元のIPアドレスがトランスレータやリバースプロキシとなる場合も

アプリケーションのIPv6対応 (2/2)
アプリケーションが依存する外部サービスのIPv6対応

- IPv6対応
  - バックエンドシステム
    - DBMS, RADIUS
  - 管理用サーバ
    - syslog, SNMP, NTP
  - X.509 CRL (Certification Revocation List)
    - クライアント証明書を利用している場合など
  - ソフトウェアアップデートサイト
    - Windows Update, Red Hat Networkなど
- 設計変更の検討が必要
  - IPv4アドレスを識別子として利用しているシステム
    - IPアドレスベースのブラックリストなど
      DNSBL, IPレピュテーションサービス
- 注意点: 匿名アドレス/一時アドレス
  - 外部サービスに接続する際に、送信元IPアドレスが変わる可能性がある

CLARA ONLINE

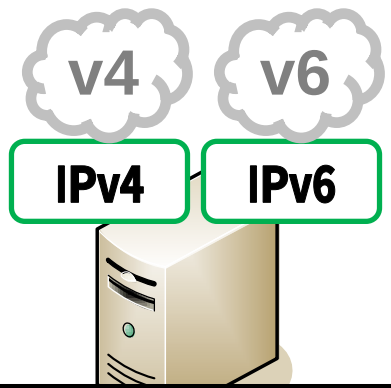
二つのデュアルスタック

サーバのデュアルスタック化と
サービスのデュアルスタック化

デュアルスタック実現の粒度

- **サーバ単位**
- **単一サーバでIPv6対応する際の課題**

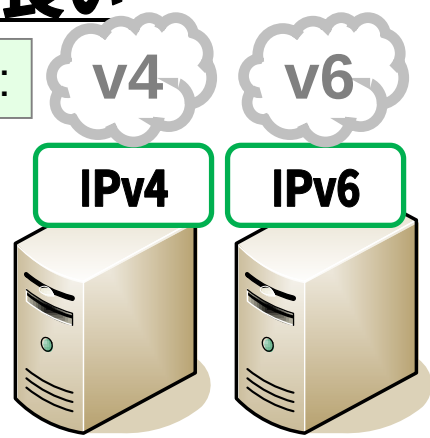
例:



	サーバ単独	システム
1台	IPv4/IPv6 デュアルスタック	N/A
複数台	IPv4またはIPv6シングルスタック (もしくはIPv4/IPv6デュアルスタック) のサーバ、もしくは中継機器との併用	

- **サービス単位**
- **複数台のサーバから構成**
  - **中継機器(middlebox)を利用する場合もある**
- **個々のサーバはシングルスタックでも良い**

例:



いずれの場合にもIPv6自体への対応が必要

他の実現手法については発表後半にて

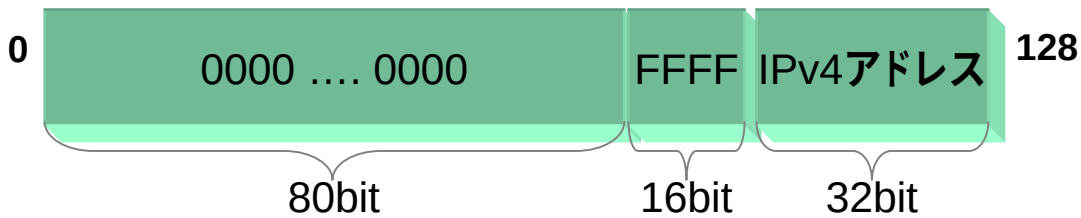
CLARA ONLINE

サーバのIPv4/IPv6デュアルスタック構成と検討課題

- IPv4 Mappedアドレスの影
- 障害発生時の切り分け

IPv4射影アドレスとサーバアプリケーション (1/4)
IPv4射影アドレス(mapped address)とは

- IPv4射影アドレスとは
  - IPv4アドレスをIPv6アドレスとして表す特殊なIPv6アドレス



  - 例: 192.0.2.128 の場合
    ::ffff:192.0.2.128 もしくは ::ffff:c000:280
  - IPv6対応アプリケーションがIPv4/IPv6対応のソケット(“::”)でIPv4のみを持つノードと通信する際に利用
    - ノード内部での利用に限定
    - 送信元･宛先アドレスとしては利用されない

IPv4射影アドレスとサーバアプリケーション (2/4)
IPv4射影アドレスを利用しない方法

- IPv4射影アドレスを利用しないようsysctl変数を設定
  - Linux (kernel 2.4.21 以降および 2.6 以降)
    - net.ipv6.bindv6only 変数を 1 に設定する
    - **デフォルトでは無効**
      (IPv4射影アドレスが利用される)
    - **一部のディストリビューションのkernelの挙動**
  - FreeBSD
    - net.inet6.ip6.v6only 変数を 1 に設定する
    - **最近のバージョンではデフォルトで有効**
      (IPv4射影アドレスは利用されない)
- アプリケーション側での対応
  - IPV6_V6ONLY ソケットオプションを利用

IPv4 射影アドレスを利用しない場合、
IPv6のソケットとは別に
別途IPv4のソケットを開く必要がある

IPv4射影アドレスとサーバアプリケーション (3/4)
ソケットの開き方: OpenSSHの場合

- IPv4/IPv6で別々にソケットをbind(2)する場合
  - “netstat –an”コマンドの結果:

sshd_configファイル:
ListenAddress 0.0.0.0
ListenAddress ::

```
tcp     0 0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp     0 0 :::22                   :::*                    LISTEN
tcp     0 0 192.0.2.2:22            10.0.0.1:34634          ESTABLISHED
```

- IPv6でのみソケットをbind(2)する場合
  - “netstat –an”コマンドの結果:

sshd_configファイル:
ListenAddress ::

```
tcp     0 0 :::22                   :::*                    LISTEN
tcp     0 0 ::ffff:192.0.2.2:22     ::ffff:10.0.0.1:34644   ESTABLISHED
```

ソケットの開き方は、TCP/IPスタックの設定、個々のアプリケーションの実装とListen設定方法により異なるので、アプリケーション毎に確認したほうがよい

IPv4射影アドレスとサーバアプリケーション (4/4)
IPv4射影アドレスとソケットの関係

- IPv4では“ 0.0.0.0”、IPv4/IPv6デュアルスタック環境では“0.0.0.0”の代わりに“::” をListenするアプリケーションでは、IPv4接続に関する挙動が変わる
  - IPv4射影アドレスが有効な場合、AF_INETソケットのIPv4アドレスではなく、AF_INET6ソケットのIPv4射影アドレスが利用される(“192.0.2.1”ではなく“::ffff:192.0.2.1”となる)
- IPv4環境では“0.0.0.0”、IPv4/IPv6デュアルスタック環境では“0.0.0.0“に加えて“::” をListenするアプリケーションでは、IPv4接続に関する挙動は変わらない

Socketの構成		IPv6クライアントからの接続	IPv4クライアントからの接続
	IPV6_V6ONLY ソケットオプション		
0.0.0.0 IN_ADDR_ANY		×	○ 192.0.2.1
:: IN6_ADDR_ANY	Yes (IPv4射影アドレスを利用しない)	○	×
	No (IPv4射影アドレスを利用する)	○	○::ffff:192.0.2.1 IPv4射影アドレス

IPv4射影アドレスを巡る議論

- Internet Draft (RFC**には至らず**)
  - IPv4-Mapped Addresses on the Wire Considered Harmful
    http://tools.ietf.org/html/draft-itojun-v6ops-v4mapped-harmful-02
  - IPv4-Mapped Address API Considered Harmful
    http://tools.ietf.org/html/draft-cmetz-v6ops-v4mapped-api-harmful-01
- IPv4-Mapped Address API Considered Harmful (draft-cmetz-v6ops-v4mapped-api-harmful-01) **の指摘**
  - **コードの移植性が低下**
  - **実装の複雑化**
  - **アクセス制御が複雑化**
    - **同じノードと通信する場合でも、**“192.0.2.128 (AF_INET)” **と** ”::ffff:192.0.2.128 (AF_INET6)” **のように見え方が異なる**

障害発生時の切り分けと設計

- 特定のプロトコルで問題が発生するケースも
  - IPv4サービスのみの障害
  - IPv6サービスのみの障害
    - Path MTU Discovery問題(後述)
    - 片方のサービスのみの障害が発生した場合、フォールバックが行われ、アクセスが遅くなるだけで気づきにくいことも
- IPv6特有のセキュリティホールバグを想定すべき
  - メンテナンス箇所の局所化
  - 例:
    - IPv6 Neighbor Discovery Protocol Neighbor Solicitation Vulnerability (CVE-2008-2476)
    - IPv6 Type 0 Routing Header Vulnerability (CVE-2007-2242)
    - Linux Kernel Netfilter nf_conntrack IPv6 Packet Reassembly Rule Bypass Vulnerability (CVE-2007-1497)
    - Linux Kernel vulnerable to DoS via the ipv6_getsockopt_sticky() function (CVE-2007-1000)

IPv4/IP6デュアルスタックサービスの実現方法の検討

- ホスティングサービスの提供方法
- End-to-Endモデル
- Middleboxモデル
- Hybridモデル

デュアルスタックサービスの実現方法

- End-to-Endモデル
  - サーバはIPv6ないしはIPv4クライアントから、直接、接続を受け付ける
  - **サーバがIPv6に対応**

- Hybridモデル
  - End-to-EndモデルとMiddleboxモデルの組み合わせ

- Middleboxモデル
  - サーバにはMiddleBox経由で接続を受け付ける
  - Middlebox:
    - IPレベルのプロトコル変換機能を備えた機器を利用
    - アプリケーションレベルゲートウェイのデュアルスタック化
  - **サーバがシングルスタックでよい**
    - IPv4だけでもよい

ホスティングサービスと提供モデル(1/2)
IPv4環境での専用サーバ/VPSサービス

- 顧客毎に{物理,仮想}サーバ全部を提供
  - 各顧客には1個以上のIPアドレスを割り当て

専用サーバ

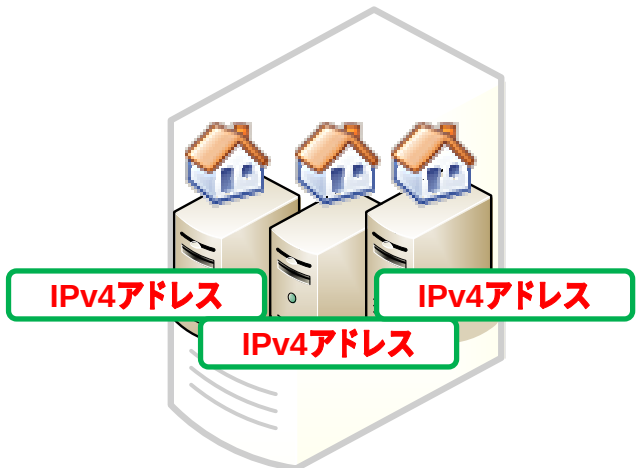
- 専用サーバ・VPS

一般的な場合

→ End-to-Endモデル

大規模なサイトの場合

→ Middleboxモデル



仮想サーバ

ホスティングサービスと提供モデル(2/2)
IPv4環境での共用サーバサービス

- 顧客にサーバ資源の一部を提供するサービス
  - 各顧客は必ずしも固有のIPアドレスを持たない
  - Named-based Virtual Hostなど、アプリケーション層の設定が識別子

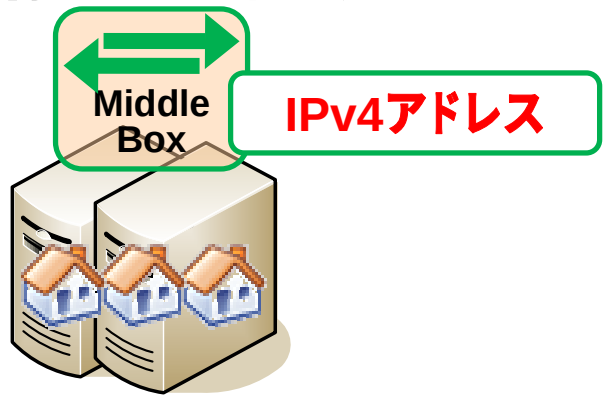
- 共用サーバ
  1. 一台ごとに独立しているシステムの場合

  →End-to-Endモデル

  2. 複数台で構成されているシステムの場合

  →Middleboxモデル

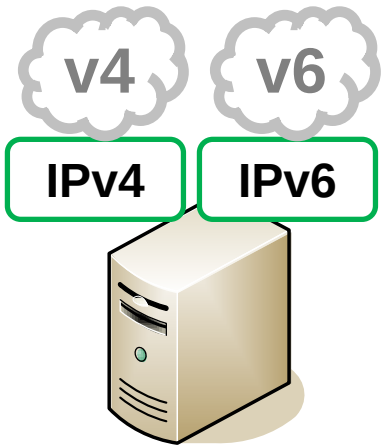
一台のサーバで構成されている共有ホスティング



複数台のサーバで構成されている共有ホスティング

End-to-EndモデルによるIPv6の導入

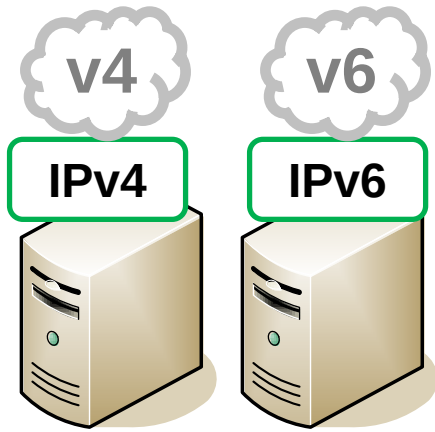
モデル1



サーバ側:

IPv4/IPv6 Dual Stack

モデル2



サーバ側:

IPv4 Single Stack
＋
IPv6 Single Stack

End-to-Endモデルによるデュアルスタックサービス

- サーバがEnd-to-Endでサービスを提供
- サーバ側はデュアルスタックで運用[モデル1]

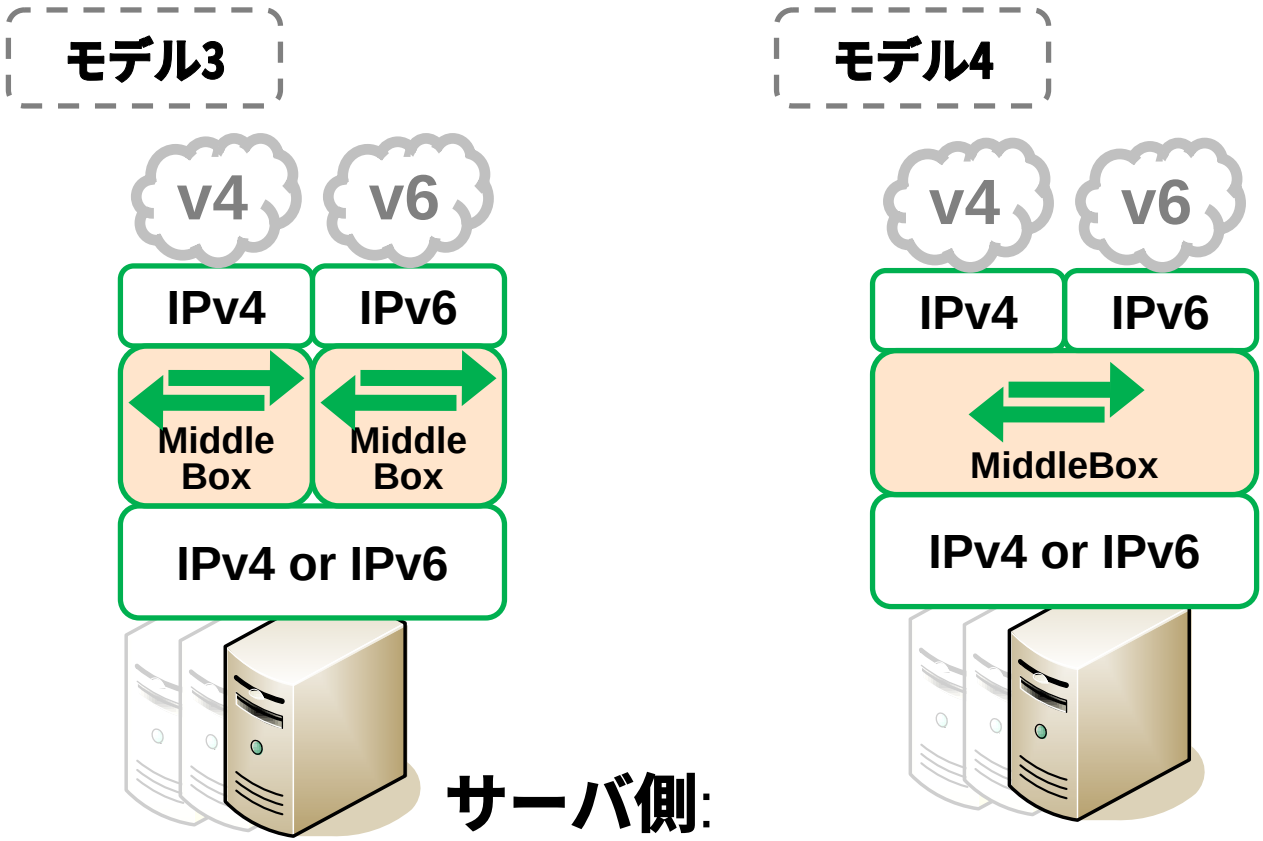
- メリット
  - 機器が最小限で済む
    - 機器コストも最小限
  - 構成がシンプル
    [モデル1]
    1台のサーバが両プロトコルのサービスを提供
    [モデル2]
    問題発生時に切り分け作業が容易
  - 新しい種類のプロトコルやアプリケーションへの対応が容易

- デメリット
  - サーバ側のIPv6対応が必須
    [モデル1]
    サーバ側でデュアルスタックに伴う問題が生じる
  - スケーラビリティに制約

Middlebox とはなにか

- RFC3234の定義
  - *“ A middlebox is defined as **any intermediary device** performing functions other than the normal, standard functions of an **IP router** on the datagram path between a source host and destination host.”*
- IPv4/IPv6プロトコル変換機能を備えた機器
  - **プロトコルトランスレータ**
  - IPv4/v6変換機能付きFirewall
  - IPv4/v6変換機能付きロードバランサ
- **アプリケーションレベルゲートウェイ**
  - **リバースプロキシ**
  - CDN (Contents Delivery Network)

IPv6の導入モデル
Middlebox



サーバ側:

IPv4 Single Stack

OR

IPv6 Single Stack

Middleboxモデルによるデュアルスタックサービス

- 全てのトラフィックはMiddleboxを経由
- サーバ側はシングルスタックで運用

- メリット
  - サーバ自体をIPv4もしくはIPv6シングルスタックで運用できる
    - サーバ側のIPv6対応は必須ではない
    - デュアルスタックでもよいが…
  - スケーラビリティを確保
    - サーバ台数を増やせる

- デメリット
  - サーバ以外の機器が必要に
    - 新たに導入する場合にはコスト増要因
    - ログ取得が複雑に
      - アクセス元のIPアドレスがMiddleboxになる場合も
  - 問題発生時に切り分け作業が複雑化
  - 今後、新たに登場するであろうプロトコルやアプリケーションの対応に制約

Middlebox: トランスレータ

- IPv4とIPv6は回線上で互換性がない
  - IPv4クライアントとIPv6シングルスタックのサーバ
  - IPv6クライアントとIPv4シングルスタックのサーバ

  プロトコルを変換して通信できるようにする必要がある
- IP層で変換を行うNAT-PT方式が主流
  - マスカレード
  - 静的マッピング(1:1)
    - 単一のIPv6アドレスを単一のIPv4アドレスに割り当て

    = IPv4アドレスの節約にはならない

    バックエンドサーバをIPv6シングルスタックにした場合でも、マッピング用のIPv4グローバルアドレスが必要に
    - サーバ向け
  - 動的マッピング(n:m)
    - DNS-ALGとの連携

Middlebox: Firewall

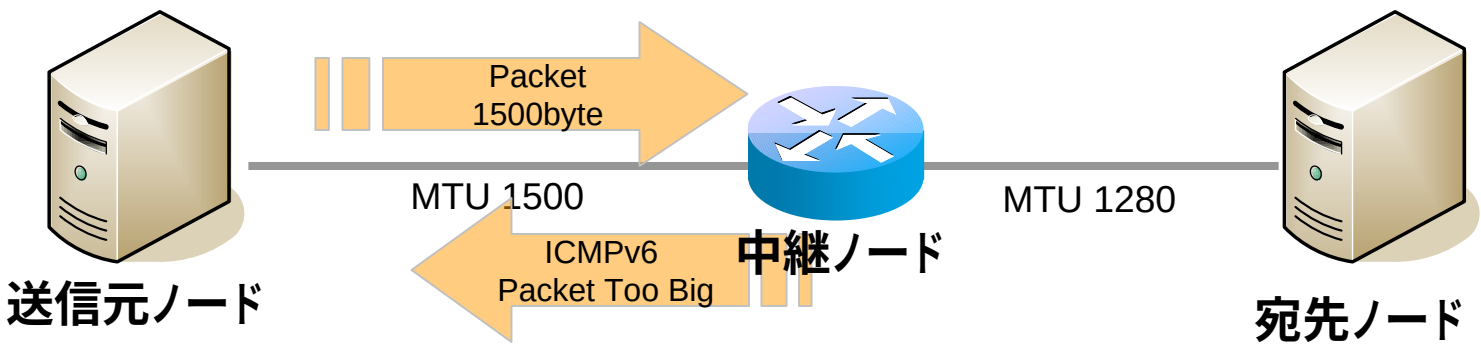
- IPv4およびIPv6のパケットをフィルタする機器
  - VPNやIPv6トンネルの終端、プロトコル変換機能を備えた製品もある
    - フィルタリングポリシーの把握が容易
- Firewallでのプロトコル変換機能の例
  - IPアドレスマッピング機能でプロトコルを変換
  - NAT内のサーバをグローバルIPアドレスで公開する

- IPv4→IPv6
  - IPv4クライアントを想定
  - フロントエンドがIPv4
  - バックエンドがIPv6

- IPv6→IPv4
  - IPv6クライアントを想定
  - フロントエンドがIPv6
  - バックエンドがIPv4

IPv6とMTU

- IPv6の最小MTUは1280バイト
- IPv6では中継ノードでパケットの分割を行わない
- Path MTU Discovery (RFC1981)
  - 相手先ノードまでの経路上で利用可能な最大MTUを調査
  - 転送しようとするパケットがルータの転送先インタフェースのMTUより大きい場合、送信元に ICMPv6 type=2 (Packet Too Big) エラーを返す
  - Firewall で ICMPv6 type=2 code=0 をフィルタしないよう注意



Middlebox: ロードバランサ

- 複数のサーバにアクセスを振り分ける機器
  - 大規模サイト向け
- NAT構成に類似
  - 仮想IPアドレスをプライベートIPv4アドレスに変換
  - DSR構成では利用できない
- 構成

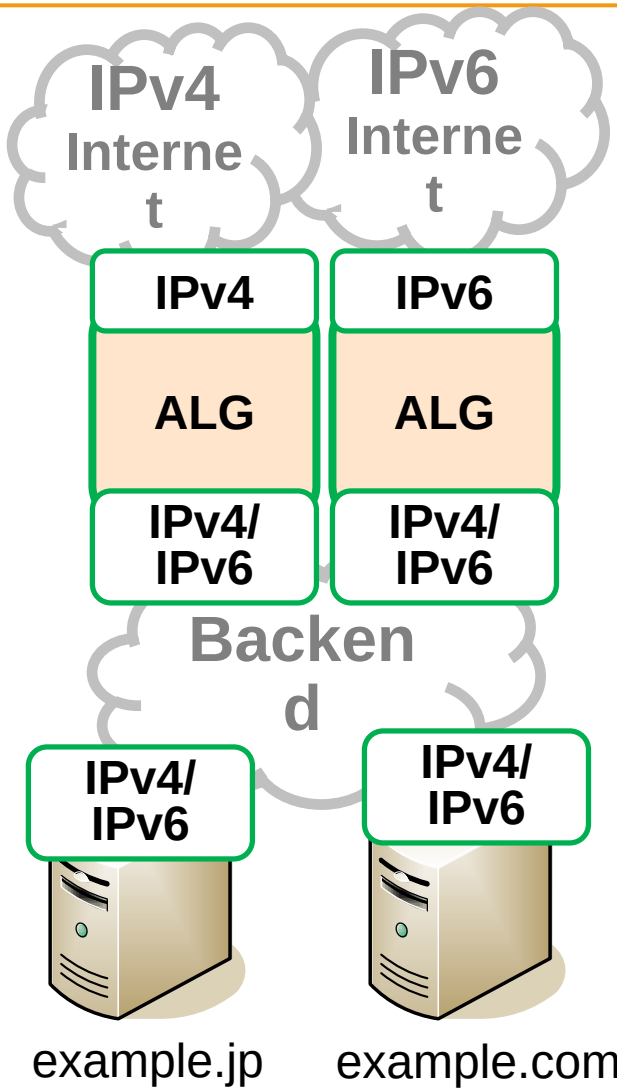
  [モデル3/モデル4]
  - IPv4/IPv6プロトコル変換を行うロードバランサのみ

  [モデル3]
  - IPv4/IPv6プロトコル変換を行うロードバランサ

  ＋

  プロトコル変換を行わないロードバランサ

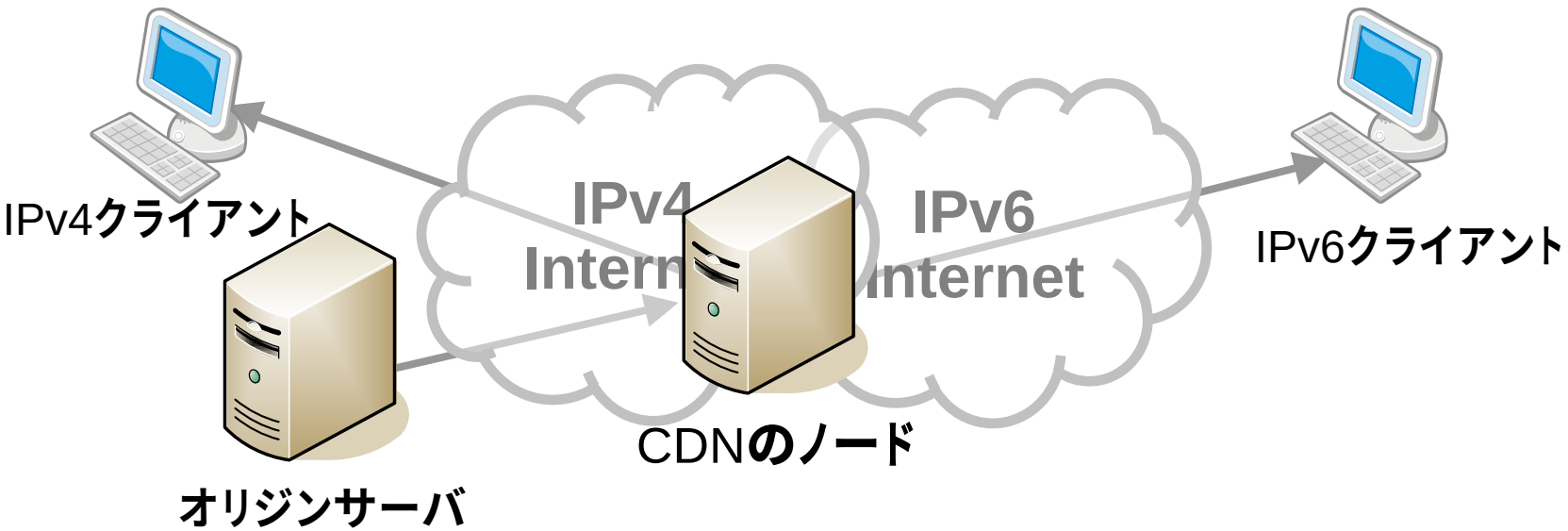
Middlebox: リバースプロキシ



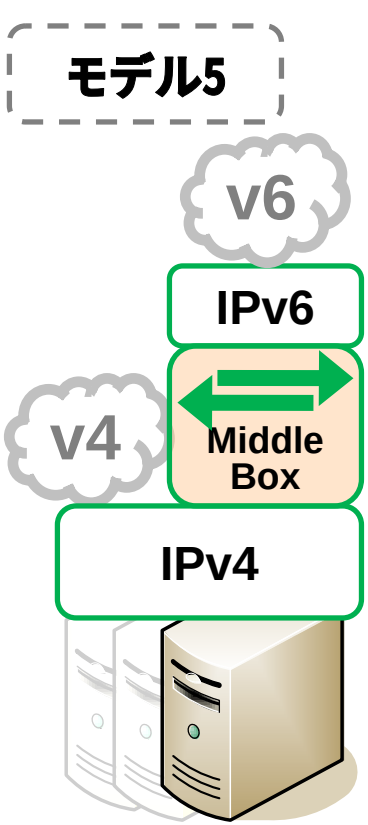
- リバースプロキシ
  - プロキシサーバのデュアルスタック運用
  - フロントエンド側ではIPv4およびIPv6でサービスを提供
  - パケットレベルではなく、アプリケーションレベルでIPv4/IPv6を変換
- メリット
  - サーバ側はシングルスタックでよい
  - アプリケーション層の機能を付加できる (例: アンチウィルス, gzip圧縮, TCP最適化)
- デメリット
  - 利用可能できるプロトコルが限定
  - 柔軟性 (例:ドメイン名の追加)
  - スケーラビリティ (例: Ajaxによる大量のセッション, Spam対策)

Middlebox: CDN

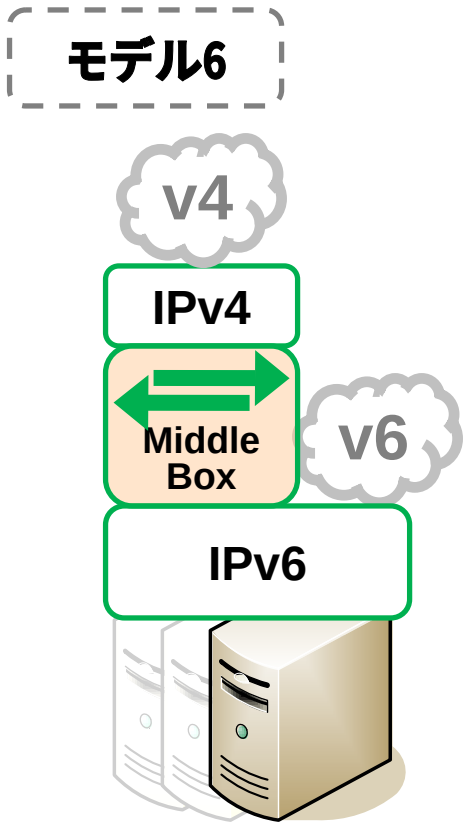
- Contents Delivery Network
  - DNSやBGP Anycastなどを利用し、アクセス元から最適なCDNのノードに誘導
  - 広域に分散した一種のリバースプロキシ
    - CDNのノードがIPv6でサービスを提供すれば、オリジンサーバはIPv4のみに対応すれば良い



IPv6の導入モデル: Hybrid Model



サーバ側:
IPv4 Single Stack



サーバ側:
IPv6 Single Stack

Hybridモデルによるデュアルスタックサービス

- サーバ側はシングルスタックで運用
- 基本的にはサーバがEnd-to-Endでサービスを提供
  - サーバ側が対応していない方のプロトコルのみをMiddleboxで変換

- メリット
  - サーバ自体をシングルスタックで運用できる
    [モデル5の場合]
    サーバ側はIPv4のみ
    IPv6対応は不要
    [モデル6の場合]
    サーバ側のIPv6のみ対応
  - Middleboxは、トラフィックの一部のみを処理

- デメリット
  - サーバ以外の機器が必要
    - コスト増要因
    - ログ取得が複雑化
  - スケーラビリティがない
  - アドホックな設計

IPv6導入時期毎の各モデルの普及予想

テスト導入期　　本格導入期

モデル1
(IPv4+IPv6 Dual Stack Server)

End-to-Endモデル

モデル2
(IPv4 Single Stack Server + IPv6 Single Stack Server)

モデル3
(IPv4 Single Stack + IPv6 Single Stack Middlebox)

Middleboxモデル

モデル4
(IPv4 + IPv6 Dual Stack Middlebox)

モデル5
(IPv4 Single Stack Sever+ IPv6 Middlebox)

Hybridモデル

モデル6
(IPv6 Single Stack Sever + IPv4 Middlebox)

デュアルスタック対応サービスの実現方法

サーバの構成	End-to-End モデル	Middlebox モデル	Hybrid モデル
IPv4/IPv6 デュアルスタック	モデル1	N/A	N/A
IPv4シングル スタック	モデル 2 他にIPv6シングル スタックのノードが必要	モデル 3,4 IPv4もIPv6も Middlebox経由	モデル 5 Middleboxが IPv6へ変換 IPv4へは直接通信
IPv6シングル スタック	モデル 2 他にIPv4シングル スタックのノードが必要	モデル 3,4 IPv4もIPv6も Middlebox経由	モデル 6 Middleboxが IPv4へ変換 IPv6へは直接通信

まとめ

- IPv6対応にあたっては各期毎に検討を
- ネットワークおよびDNSのIPv6対応が必須
- サーバとサービスのデュアルスタック化は異なる
  - サーバのデュアルスタック化には特有の課題がある
  - シングルスタックのサーバを組み合わせてサービス全体をデュアルスタック化する手法もある
- サービスのデュアルスタック化には、様々な実現方法が考えられる

CLARA ONLINE

Thank you!