クライアントOSのIPv6実装事情

株式会社インテック・ネットコア
ネットワークプラットフォーム研究開発グループ
北口善明

目次

- クライアントOSのIPv6対応状況
- IPv6対応OSにおける挙動整理
  - DNSリゾルバの実装
  - アドレス選択機構の実装
  - 自動トンネリングの実装
- まとめ

クライアントOSのIPv6対応状況

2009年のクライアントOS

- ほぼ全てのクライアントOSはIPv6 Ready!

OS	Ready？	備考
Windows XP	Yes	デフォルトではIPv6は無効[注] DNSリゾルバは非対応
Windows Vista	Yes	WindowsのIPv6正式サポート版
Windows 7 RC	Yes	Vistaと同様の実装
Mac OS X	Yes	Jaguarから正式サポート FreeBSD 5系のIPv6実装
Linux	Yes	Kernel-2.2系からサポート
BSD	Yes	KAMEプロジェクトの実装
Solaris	Yes	Solaris 8からサポート

注Windows XPにおける「ipv6」コマンドは、ベータ版からの古いコマンドであるため現在は利用を推奨されていない。
IPv6を有効化するには次の方法が推奨される。

1. GUI利用
   コントロールパネル＞ネットワーク接続＞インターフェース　（プロパティからプロトコルを追加する）
2. netshコマンド
   「netsh interface ipv6 install」

Windows Vista

- 代表的なコンシューマOSがIPv6に完全対応
  - IPv6がデフォルトで有効
  - GUIによるIPv6設定
  - IPv4/IPv6を意識させないAPI
- ほとんどのWindowsコンポーネントがIPv6対応
  - IPv6 onlyは容易だがIPv4 onlyは基本的に不可
- 自動トンネリング機能
  - IPv6へ到達可能なトンネル接続を自動的に実施

IPv6対応OSにおける挙動整理

①DNSリゾルバの実装

DNSのIPv6対応

- DNSのIPv6対応が持つ二つの意味
  - RR（リソースレコード）のIPv6対応
    - ◆AAAA RRによる正引き登録
    - ◆ip6.arpaドメインを用いたPTR RRによる逆引き登録
  - トランスポートのIPv6対応
    - ◆DNS通信のIPv6利用
- IPv6（デュアルスタック）時代の名前解決
  - A RRとAAAA RRの両方を利用する
    - ◆順次問い合わせを行う（実装により順番が異なる）
    - ◆クエリ数はAクエリ＋AAAAクエリなので単純に二倍に増加
  - 名前解決と利用プロトコルは独立
    - ◆AAAAクエリをIPv4通信で可能

DNSリゾルバ実装

- Aクエリを優先する実装が一般的
  - AAAAクエリに未対応な機器による問題を回避するため

◆壊れた応答例（RFC4707）

① AAAAレコードの問い合わせを無視	② NXDOMAIN（RCODE=3）を返す
③ NXDOMAIN以外の不正なRCODEを返す	④ 壊れた返答/IPv4アドレスを返す
⑤ Lame Delegationになる	

※正しい応答はNOERROR（RCODE=0）で中身が空

  - Aクエリの応答時間を基にAAAAクエリの待ち時間を決定（FreeBSD，Vistaなど）
- AAAAクエリの抑制（Windows Vista）
  - AクエリでNXDOMAINならAAAAクエリを抑制
  - グローバルIPv6アドレス※が付与されない限りAAAAクエリを抑制

※ Teredoアドレスを除くグローバルIPv6アドレス

OS毎のDNSリゾルバ実装

- クエリ順序はOSで異なる
  - AAAAクエリを先に実施するOS
    - ◆Windows XP，Linux
  - Aクエリを先に実施するOS
    - ◆Windows Vista，Windows 7 RC，FreeBSD，Mac OS X
- 利用プロトコルの優先順位
  - IPv6を優先的に利用するOS
    - ◆Windows Vista，Windows 7 RC
  - IPv4しか利用できないOS
    - ◆Windows XP
  - 設定ファイルに依存するOS（/etc/resolv.confの順序）
    - ◆FreeBSD，Linux

（おまけ）DNSディスカバリ

- DNSサーバアドレス取得方法に３種類の手法
  - RAによる通知（RDNSSオプション） RFC5006 (Experimental)
  - DHCPv6による通知 RFC3315 実装あり
  - Well-known Anycast Addressの利用
    - ◆サイトローカル利用に問題（XP，Vistaには設定が残っている）
- DHCPv6による設定が現状一般的
  - RAのOフラグとMフラグによりDHCPv6に移行可能
    - ◆ただしRFC4862では削除された仕様
  - Windows Vistaでは
    - ◆Oフラグ設定でステートレスDHCPv6が動作
      - ❖DNSサーバなどのネットワークパラメータのみ取得
    - ◆Mフラグ設定でステートフルDHCPv6が動作
      - ❖IPv6アドレスもDHCPv6で設定

IPv6対応OSにおける挙動整理

②アドレス選択機構の実装

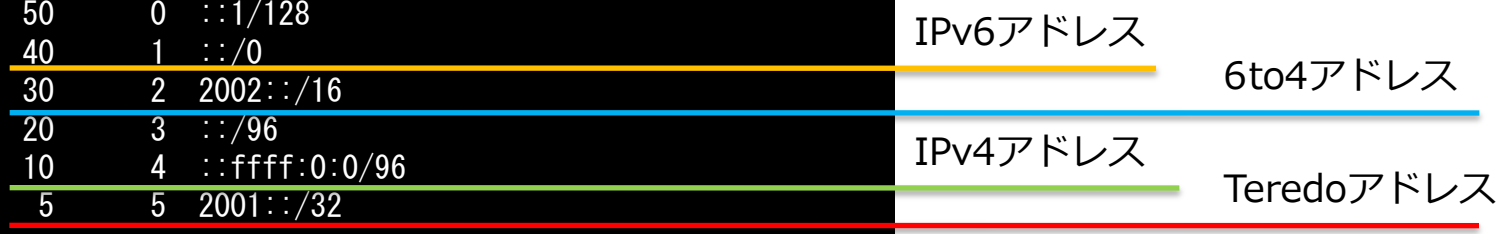
アドレス選択機構（RFC3484）

- IPv6では複数のアドレスを使い分ける必要がある
  - リンクローカルアドレスとグローバルアドレス
  - IPv4アドレスとIPv6アドレス　など
- ポリシーテーブル
  - アドレス選択時に利用するラベルや優先度を定義
  - 優先度：終点アドレス選択時に利用され高い値ほど優先
  - ラベル：始点/終点アドレス選択時に利用され一致するものを優先
- 実装状況
  - RFC3484の実装はほぼすべてのOSで完了
  - ポリシーテーブル操作が不可のものもある
    - ◆Linuxはkernel 2.6.25（要iproute2-2.6.25以上）から送信元アドレスのためのラベル操作が可能

ポリシーテーブルの実装（1）

- Windows Vistaのデフォルト設定
  - **netsh interface ipv6 show prefixpolicies** で確認可能

```
C:>netsh interface ipv6 show prefixpolicies
アクティブ状態を照会しています...

優先順位    ラベル   プレフィックス
----------  -----  ------------------------------
        50      0  ::1/128
        40      1  ::/0
        30      2  2002::/16
        20      3  ::/96
        10      4  ::ffff:0:0/96
         5      5  2001::/32


C:¥Windows¥system32>
```



  - RFC3484における標準設定＋Teredoアドレス

ポリシーテーブルの実装（2）

- Linux（kernel 2.6.25以上）
  - **ip addrlabel show** で確認可能（iproute2-2.6.25以上）

```
$ ip addrlabel show
prefix ::1/128 label 0
prefix ::/96 label 3
prefix ::ffff:0.0.0.0/96 label 4
prefix 2001::/32 label 6
prefix 2002::/16 label 2
prefix fc00::/7 label 5
prefix ::/0 label 1
```

ULA

  - LinuxではULAのラベルも独自追加されている
  - 優先度は/etc/gai.confで設定

```
$ cat /etc/gai.conf
...
#precedence ::1/128 50
#precedence ::/0 40
#precedence 2002::/16 30
#precedence ::/96 20
#precedence ::ffff:0:0/96 10
...
```

※デフォルトは/usr/share/doc/glibc-common-2.*/gai.conf

ポリシーテーブルの実装（3）

- FreeBSD
  - **ip6addrctl show** で確認可能

```
% ip6addrctl show
Prefix                    Prec Label    Use
::1/128                     50     0      0
::/0                        40     1     10
2002::/16                   30     2      0
::/96                       20     3      0
::ffff:0.0.0.0/96           10     4      0
```

※6.2 RELEASE, 7.0 RELEASEまでは
/etc/rc.confに
ip6addrctl_enable="YES“
の設定が必要

  - RFC3484の標準設定のみ

ポリシーテーブルの応用

- IPv4優先設定
  - デフォルトのポリシーテーブルはIPv6の優先度が高い

  ➡ IPv4優先は“::ffff:0:0/96”の優先度を高くするとOK

◆Windows Vista／XP

```
C:>netsh interface ipv6 set prefixpolicies ::ffff:0:0/96 45 4
C:>netsh interface ipv6 add prefixpolicies ::1/128 50 0
C:>netsh interface ipv6 add prefixpolicies ::/0 40 1
C:>netsh interface ipv6 add prefixpolicies 2002::/16 30 2
C:>netsh interface ipv6 add prefixpolicies ::/96 20 3
C:>netsh interface ipv6 add prefixpolicies 2001::/32 5 5
```

※初めての設定時には全てのエントリが消えるので追加が必要
Windows XPの場合には**add**ではなく**set**となる

◆Linux

```
# vi /etc/gai.conf
…
precedence ::ffff:0:0/96 45
…
```

※/etc/gai.confに変更行のみ追加
::/0の優先度はデフォルトの40とした場合

◆FreeBSD

```
# /etc/rc.d/ip6addrctl prefer_ipv4
```

※優先度を切り替えるスクリプトがある

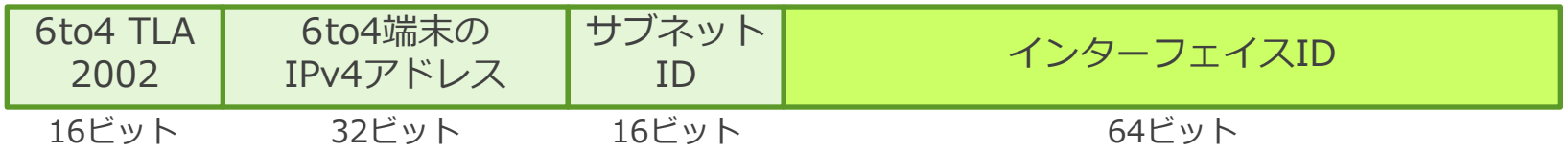
IPv6対応OSにおける挙動整理

③自動トンネリングの実装

自動トンネリング

- 6to4（RFC3056）
  - トンネル接続とIPv6アドレス割り当てを同時に実現
  - IPv4グローバルアドレスを利用したIPv6アドレス

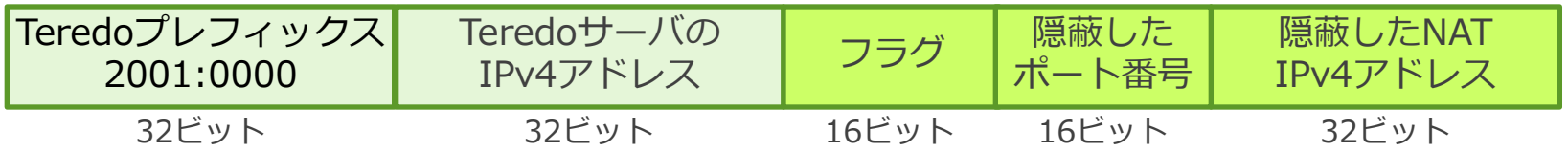
◆6to4のアドレス形式



・/48のアドレス空間が割り当てられる

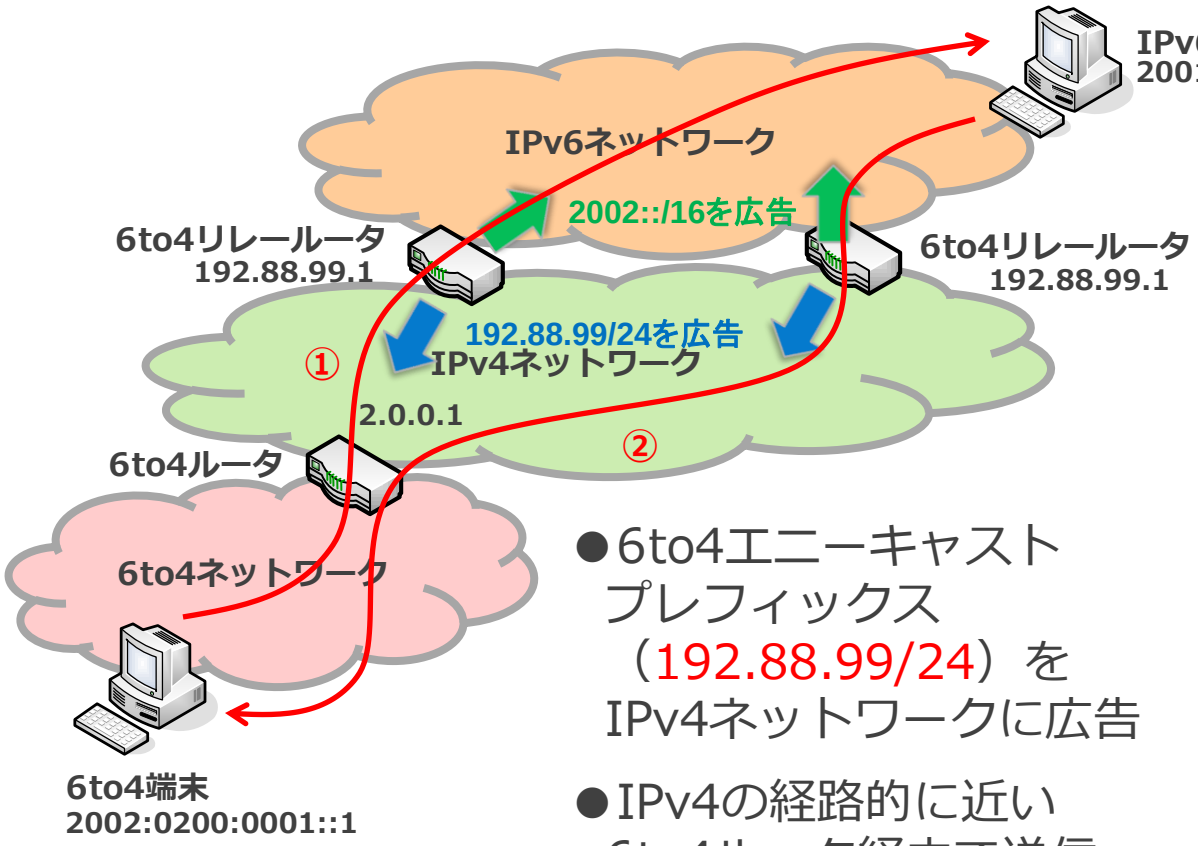
- Teredo（RFC4380）
  - NATトラバーサルをIPv6で実現する技術
  - NATの内側からIPv6トンネル接続が可能

◆Teredoのアドレス形式



・/128のアドレスが一つ割り当てられる

6to4端末⇔IPv6端末通信の仕組み



- 6to4エニーキャストプレフィックス（192.88.99/24）をIPv4ネットワークに広告
- IPv4の経路的に近い6to4ルータ経由で送信（①）
- 6to4空間（2002::/16）をIPv6ネットワークに広告
- IPv6の経路的に近い6to4ルータ経由で返信（②）

往復の経路は基本的に異なる

⇩

トラブルシュートが困難

※6to4リレールータはボランティア運用が一般的

6to4の実装と挙動

- Windows Vistaでは
  - IPv4グローバルアドレスが設定されると6to4トンネルインターフェイスもデフォルトで設定する

```
C:>ipconfig
...
Tunnel adapter ローカル エリア接続* 20:

   接続固有の DNS サフィックス . . . :
   IPv6 アドレス . . . . . . . . . . . : 2002:dxxx:xxx2::dxxx:xxx2
   デフォルト ゲートウェイ . . . . . : 2002:c058:6301::c058:6301
...
```

※6to4リレールータとして**6to4.ipv6.microsoft.com**（192.99.88.1）が設定されている

- 利用されるのはIPv6オンリーサーバのみ
  - ポリシーテーブルのラベルがIPv6アドレスと異なる
  - デュアルスタックサーバに対しては
    - ◆ラベルが一致するIPv4アドレスが選ばれるため6to4は使われない

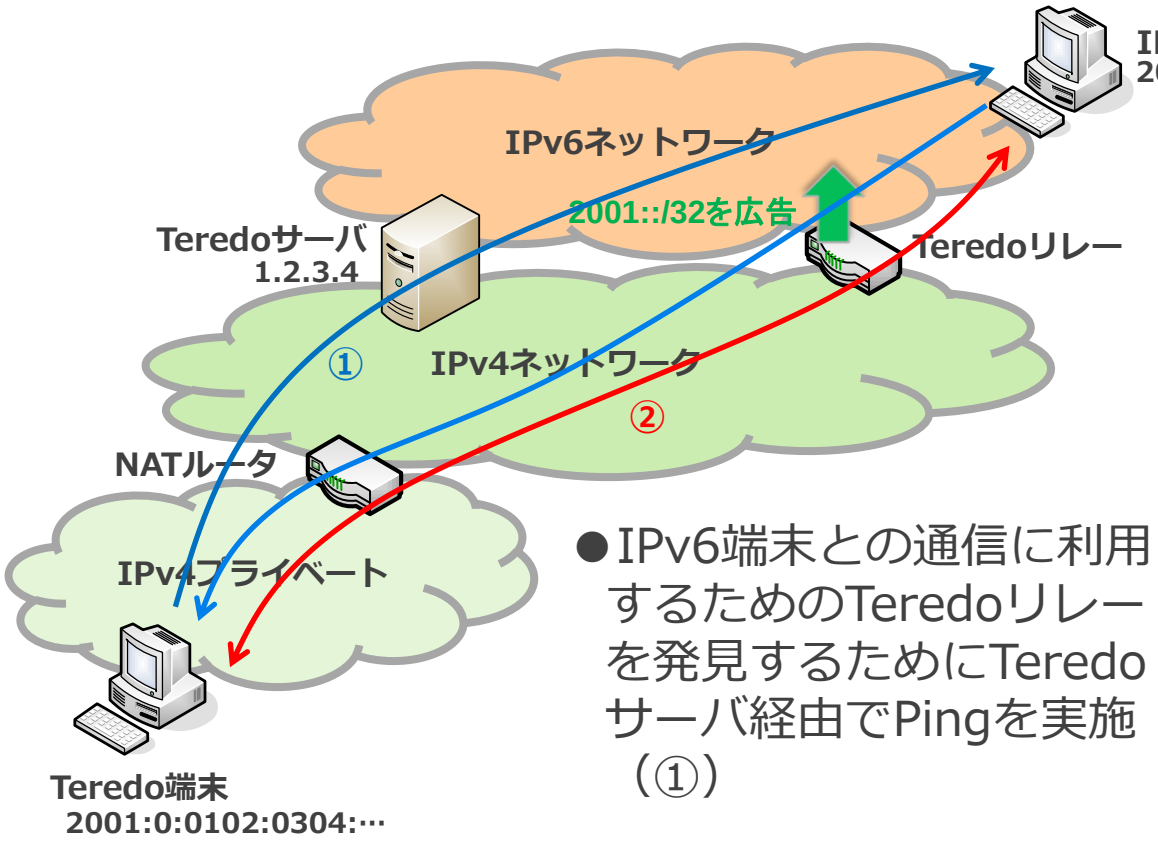
（おまけ）6to4利用設定

- デュアルスタックサーバに6to4で接続する方法
  - IPv6アドレスと6to4アドレスのラベルを合わせると良い
  - netsh interface ipv6 set prefixpolicies 2002::/16 30 1

```
C:>netsh interface ipv6 show prefixpolicies
アクティブ状態を照会しています...

優先順位    ラベル   プレフィックス
----------  -----  --------------------------------
        50      0  ::1/128
        40      1  ::/0
        30      1  2002::/16
        20      3  ::/96
        10      4  ::ffff:0:0/96
         5      5  2001::/32


C:¥Windows¥system32>
```

※初めての設定時には全てのエントリが消えるので追加が必要

Teredo端末⇔IPv6端末通信の仕組み



- IPv6端末との通信に利用するためのTeredoリレーを発見するためにTeredoサーバ経由でPingを実施（①）

- Teredo空間（2001::/32）をIPv6ネットワークに広告
- IPv6の経路的に近いTeredoリレー経由で返信

IPv6端末に近いTeredoリレーを経由して通信（②）

※TeresoサーバやTeredoリレーはボランティア運用が一般的

Teredoの実装と挙動

- Windows Vistaでは
  - NATルータ配下のIPv4プライベートアドレスが設定されるとTeredoトンネルインターフェイスをデフォルトで設定する（Windows Firewallが有効である場合）

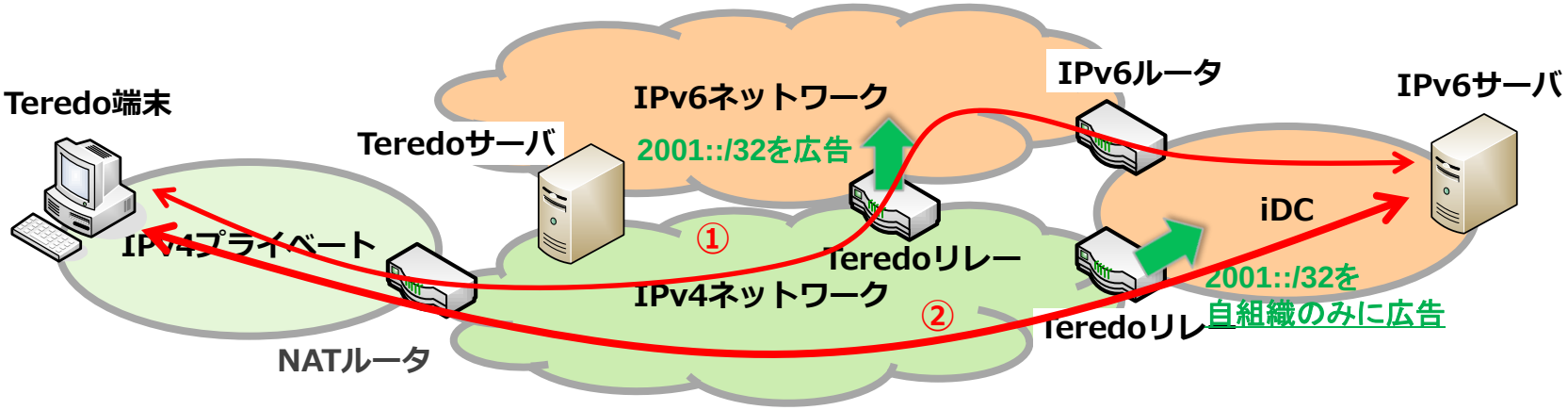
```
C:>ipconfig
...
Tunnel adapter ローカル エリア接続* 6:

   接続固有の DNS サフィックス . . . :
   IPv6 アドレス . . . . . . . . . . . : 2001:0:cf2e:3096:247d:XXXX:XXXX:YYYY
   リンクローカル IPv6 アドレス. . . . : fe80::247d:XXXX:86fd:XXXX%10
   デフォルト ゲートウェイ . . . . . :
...
```

※Teredoサーバとして**teredo.ipv6.microsoft.com**が設定されている

- Teredoアドレスのみではは利用されない
  - AAAAクエリを実施しないので実質利用されない
    - ◆IPv6アドレスを指定すると通信を行う
  - 中から一度発信しないと外から受け付けない

Teredoの経路制御

- サーバ（iDC）側でコントロール可能
  - Teredoリレーをサーバ隣に設置すると．．．



- 通常はネットワーク上でIPv6的に近い ”他組織が提供する” Teredo リレーを経路制御で選ばれて利用される（①）
- 自組織で提供する Teredoリレーを設置することで利用するTeredo リレーをコントロールできる（②）

TeredoによるIPv6通信をIPv4ネットワーク主体で利用可能

まとめ

まとめ

- クライアントはIPv6 Ready！
  - IPv6が機能している認識が必要
    - ◆IPv4ネットワークでもIPv6名前解決が行われたり
    - ◆IPv4を経由したIPv6通信能力を備えていたり
  - ネットワーク／サービスのIPv6対応待ち

- IPv6 Ready＝デュアルスタック
  - IPv4のみの場合と挙動が複雑
    - ◆運用者はこの挙動を理解しておく必要がある
  - 自動トンネル機能はIPv6世界への近道だがトラブルシュートが複雑
    - ◆リレールータは他組織のボランティア運用で原因特定が困難