IPv6 の概要と 6bone の動向

奈良先端科学技術大学院大学
山本和彦
Kazu@Mew.org

内容

IPv6
- 歴史、設計思想、アドレス・アーキテクチャ、近隣探索

IPsec
- 歴史、設計思想、認証、暗号

6bone
- 歴史、現状

IPv6 の現状

IETF の IPng 分科会
- http://playground.sun.com/pub/ipng/html/ipng-main.html

ルータ
- Bay Networks、Cisco、3Com

ホスト & ルータ
- 日立、東芝、富士通、
- WIDE プロジェクト、INRIA、
- NRL、UNH、etc...

次世代 IP の必要性

経路表増加の抑制
- 1組織1クラスBアドレスの割り当て

クラスBアドレスの枯渇
- 複数のクラスCアドレスの割り当て

経路表の急増
- CIDR による経路表増加の抑制

それでもインターネットは成長する
- IPv4アドレス全体の枯渇

アドレス空間の大きな IP が必要
- 次世代 IP or IPng(IP next generation)

OSI or the Internet?

1992年6月 INET 97@神戸
- IPng の模索

IAB の「CLNPでいくぞ」発言
- IAB はインターネットを OSI に売った？

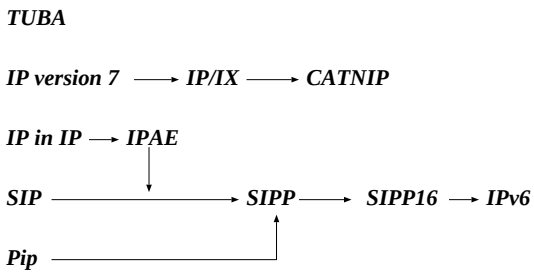
IAB の解散

仕切り直し

IAB: Internet Activity Board → Internet Architecture Board
CLNP: Connection-Less Network Protocol
OSI: Open Systems Interconnection

2 年間の競合

1992年 ～ 1994年



TUBA: TCP and UDP over Bigger Addresses
IPAE: IP Address Encapsulation
SIP: Simple IP
SIPP: Simple IP Plus
(注) version 5 は ST が使っている

IPv6 の設計思想

IPv4 よりも大きなアドレス空間を提供
IPv4 の長所を引き継ぐ
IPv4 よりも効率をよくする
2代目のジレンマを避ける
- 多機能を追求して仕様を太らせない

だれでも使えるように
セキュリティ機能の導入

IPv6 の特徴

アドレスの拡張
- 2^32 = 43 億 → 2^128 = 3.4 × 10^38

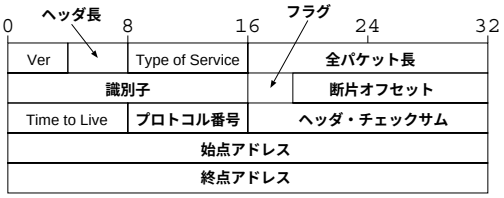
ヘッダの簡略化
- ヘッダ長、TOS、断片オフセットなどの排除

数珠つなぎヘッダ
- 利用頻度の低い機能を追い出す(断片ヘッダなど)
- 汎用的なオプションの定義

プラグ&プレイ
- デフォルト経路、プレフィックスなどの取得

セキュリティ
- IPsec が必須

IPv4 ヘッダ



除去:

- ヘッダ長、すべてのオプション (ヘッダ長の固定化)
- TOS、ヘッダチェックサム
- 識別子、フラグ、断片オフセット (できるだけ断片化しない)

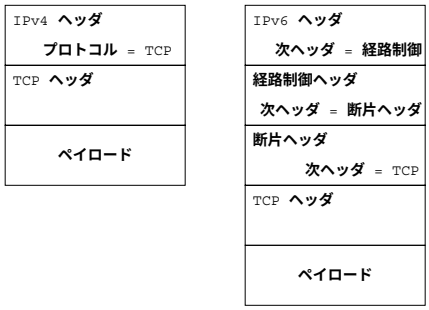
名称変更:

- プロトコル番号 → 次ヘッダ番号
- TTL → 中継限界数 (Hop Limit)

IPv6 ヘッダ

- アドレス長は 4 倍、ヘッダ長は 2 倍
- オプションは拡張ヘッダで実現
- 中継限界数は最大 255

拡張ヘッダの数珠つなぎ



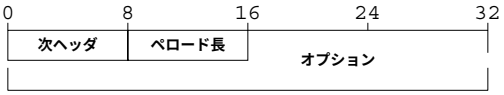
- TCP や UDP を示すプロトコル番号を次ヘッダに抽象化
- 汎用的なオプション
- TCP、UDP、認証ヘッダ、暗号ペイロードの IPv4 との共有

拡張ヘッダとヘッダ番号

0 中継点オプション・ヘッダ (Hop-by-Hop Options Header)
1 ICMP
4 IPv4 ヘッダ
6 TCP ヘッダ
13 UDP ヘッダ
41 IPv6 ヘッダ
43 経路制御ヘッダ (Routing Header)
44 断片ヘッダ (Fragment Header)
50 暗号ペイロード <IPsec>
51 認証ヘッダ <IPsec>
58 ICMPv6
60 終点オプション・ヘッダ (Destination Options Header)

ICMP: Internet Control Message Protocol

オプション・ヘッダ



「オプション番号、長さ、値」 形式

8 ビットのオプション番号

- ICMP 動作ビット
- change en-route ビット (for IPsec)

中継点オプション・ヘッダ

- 巨大ペイロード・オプション

終点オプション・ヘッダ

アドレスの表記

16 ビットの 16 進数を ":" で区切る

- 3ffe:0501:0008:0000:2060:97ff:fe40:efab
- ff02:0000:0000:0000:0000:0000:0000:0001

それぞれの先頭の 0 は省略可

- 3ffe:501:8:0:2060:97ff:fe40:efab
- ff02:0:0:0:0:0:0:1

連続する 0 は "::" で表現可

- 3ffe:501:8::2060:97ff:fe40:efab
- ff02::1

プレフィックス長は "/" の後に (0 〜 128)

- 3ffe:100::/16

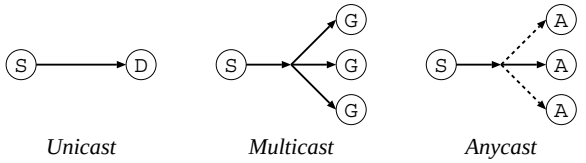
アドレスの種類

ユニキャスト
- 特定の 1 ホストと通信

マルチキャスト
- ホストのグループと通信

ブロードキャスト
- あるリンクに属す全ホストと通信

エニーキャスト
- 複数のホストが受け取れるアドレスに送信、受け取るのは 1 ホスト



アドレスのおおまかな分類

3 ビットのプレフィックス (2進数)

000 特殊なアドレス
001 経路集約型アドレス
010 未割り当て (was プロバイダ型アドレス)
011 未割り当て (was 地域型アドレス)
100 未割り当て
101 未割り当て
110 未割り当て
111 リンクローカル、サイトローカル、マルチキャスト

(注) ブロードキャストは無くなった

特殊なアドレス (ユニキャスト)

ループバック・アドレス

- 0000:0000:0000:0000:0000:0000:0000:0001 or ::1

未指定アドレス

- 0000:0000:0000:0000:0000:0000:0000:0000 or ::
- 重複アドレス検知に利用

IPv4 互換アドレス

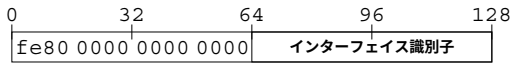
- 0000:0000:0000:0000:0000:0000:xxxx:xxxx
- (例) ::163.221.202.11
- 自動トンネルに利用

IPv4 射影アドレス

- 0000:0000:0000:0000:0000:ffff:xxxx:xxxx
- (例) ::ffff:163.221.202.11
- カーネルの実装に利用

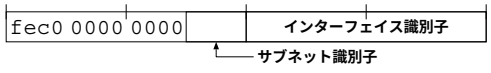
ローカルアドレス

リンクローカル



- (例) fe80::2060:97ff:fe40:efab

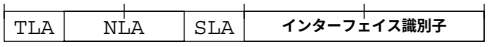
サイトローカル



- (例) fec0::1234:2060:97ff:fe40:efab

グローバル・アドレス

経路集約型アドレス
- 位置情報と識別子の分離
- パブリックとプライベートの分離



TLA (Top Level Aggregator)
- 8,192 個

NLA (Next Level Aggregator)
- NLA1, NLA2,...

SLA (Site Level Aggregator)
- サイトローカルと共有
- (例) 3ffe:501:8::2060:97ff:fe40:efab

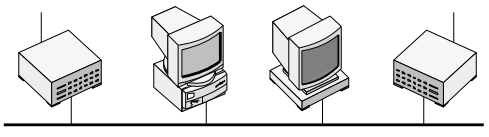
エニーキャスト

サービス探索に利用

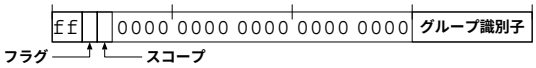
みかけ上はユニキャストと区別が付かない

あまり経験がない
- 定義されているのはサブネット・ルータ・エニーキャストのみ

だれが受け取るのか？
- サブネット外からは経路制御で決まる
- サブネット内からは近隣探索で決まる



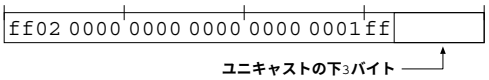
マルチキャスト



4ビットのスコープ
- 1 ノードローカル・スコープ
- 2 リンクローカル・スコープ
- 5 サイトローカル・スコープ
- 8 組織ローカル・スコープ
- e グローバル・スコープ

32 ビットのグループ識別子
- ff01::1 (ノードローカル全ノード)
- ff02::1 (リンクローカル全ノード)
- ff02::2 (リンクローカル全ルータ)

要請マルチキャスト・アドレス



アドレス解決(いわゆる ARP)に使う
- ブロードキャストはない
- リンクローカル・全ノード・アドレスは大きすぎる
- もう少し小さいマルチキャスト・アドレスが必要
- (例) fe80::2056:01ff:fe12:3456 → ff02::1:ff12:3456
- 通常のマルチキャストでは ff00:0000 〜 ffff:ffff を使わない

イーサネット・アドレスと IPv6 アドレス

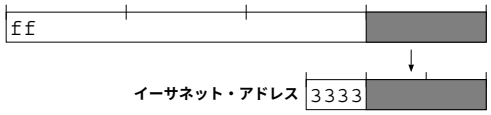
リンクローカル

- fe80::2060:97ff:fe40:efab ← 00:60:97:40:ef:ab

経路集約型アドレス (グローバル)

- 3ffe:501:808::2060:97ff:fe40:efab ← 00:60:97:40:ef:ab

マルチキャスト



- ff02::1 → 33:33:00:00:00:01
- ff02::1:ff40:efab → 33:33:ff:40:ef:ab

DNS

正引き

- AAAA レコード
- (例) mine.v6.org. IN AAAA 3ffe:501:808:1:200:f8ff:fe01:6317

逆引き

- PTR レコード
- (例)
- $ORIGIN 1.0.0.0.8.0.8.0.1.0.5.0.e.f.f.3.IP6.INT.
- 7.1.3.6.1.0.e.f.f.f.8.f.0.0.2.0 IN PTR mine.v6.org.

BIND 4.9.4 以降でサポート

要求にはまだ IPv4 を使う

API (1)

```
 struct sockaddr {
     u_char sa_len;    /* 16 bytes */
     u_char sa_family;    /* address family */
     char sa_data[14];    /* actually longer; address value */
};
struct sockaddr_in {
     u_char sin_len;    /* 16 bytes */
     u_char sin_family;    /* AF_INET */
     u_int16m_t sin_port;    /* AF_INET6 */
     struct in_addr sin_addr;    /* IPv4 address */
     char sin_zero[8];    /* padding */
};
struct sockaddr_in6 {
     u_char sin6_len;    /* 24 bytes */
     u_char sin6_family;    /* AF_INET6 */
     u_int16m_t sin6_port;    /* transport layer port # */
     u_int32m_t sin6_flowinfo;     /* IPv6 flow information */
     struct in6_addr sin6_addr;    /* IPv6 address */
 };
```

API (2)

IPv4

- 正引き struct hostnet *gethostbyname(const char *name)
- 逆引き struct hostnet *gethostbyaddr(const char *addr, int len, int af)

IPv6

- 正引き int getaddrinfo(const char *hostname, const char *servname,
  const struct addrinfo *hints,
  struct addrinfo **res);
- 逆引き int getnameinfo(const struct sockaddr *sa, size_t salen,
  char *host, size_t hostlen,
  char *serv, size_t servlen,
  int flags);

struct hostnet *gethostbyname2(char *name, int af)

ICMPv6

1 終点到達不能 (Destination Unreachable)
2 パケット過大 (Packet Too Big)
3 時間超過 (Time Exceeded)
4 パラメータ問題 (Parameter Problem)
128 エコー要求 (Echo Request)
129 エコー返答 (Echo Reply)
133 ルータ要請 (Router Solicitation)
134 ルータ通知 (Router Advertisement)
135 近隣ホスト要請 (Neighbor Solicitation)
136 近隣ホスト通知 (Neighbor Advertisement)
137 向け直し (Redirect)

近隣探索 (プラグ&プレイ)

アドレスの自動生成
重複アドレス検知 (Duplicate Address Detection)
デフォルト経路の取得
プレフィックスの取得
アドレス解決 (Address Resolution)
到達不能検知 (Neighbor Unreachability Detection)
向け直し (Redirect)

アドレスの自動生成

イーサネット (IEEE 802 アドレス)
- 00:60:97:40:ef:ab

インターフェイス識別子 (EUI 64 アドレス) の生成
- 2060:97ff:fe40:efab

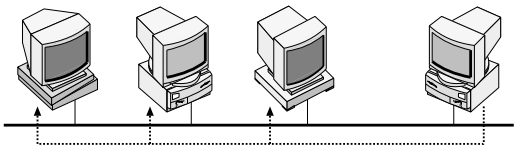
リンクローカルを仮に割り当てる
- fe80::2060:97ff:fe40:efab

マルチキャスト・アドレスへの参加

リンクローカル・全ノード・マルチキャスト・アドレス
- ff02::1

要請マルチキャスト・アドレス
- fe80::2060:97ff:fe40:efab → ff02::1:fe40:efab

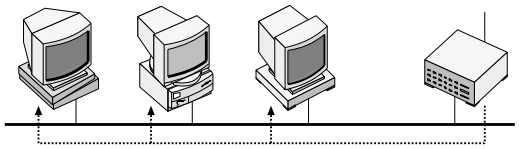
重複アドレス検知



近隣要請を出す
- 終点アドレスは、要請マルチキャスト・アドレス
- 始点アドレスは、未指定アドレス (::)
- 対象アドレスは、自分の仮のアドレス

対象アドレスが重複した場合
- 近隣通知で重複を知らせる
- 終点アドレスはリンクローカル全ノード・マルチキャスト

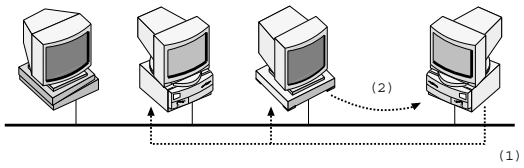
デフォルト経路とプレフィックスの取得



ルータ通知
- 定期的なアナウンス (to ff02::1)
- ルータ要請(to ff02::2) への反応
- デフォルト経路、プレフィックス、etc...

グローバル・アドレスの生成
- プレフィックス＋インターフェイス識別子
- (例)3ffe:0501:0808::2060:97ff:fe40:efab

アドレス解決

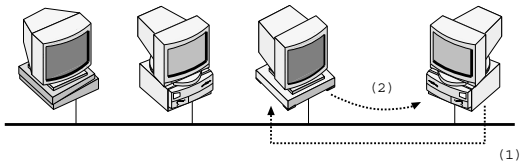


ARP の抽象化
- IPv4 と違いデータリンクごとに ARP を定める必要はない

近隣要請
- 近隣要請マルチキャスト・アドレスに近隣要請

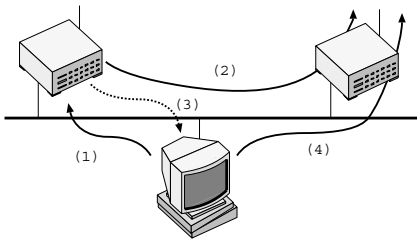
近隣通知
- 受信ホストは対象アドレスを検査
- 対象アドレスが一致すると MAC アドレスを応答

到達不能検知



近隣キャッシュ(ARP表)は状態を持つ
ユニキャストの近隣要請
- 時間がたって「あやしく」なったとき

向け直し



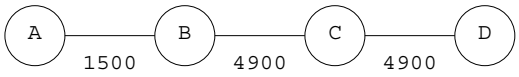
- ホストは経路制御に参加しない
- ホストははじめデフォルト経路のみを持つ
- 間違っている経路は向け直してもらい、学習する
- 次からは正しいルータに転送できる

経路 MTU 探索

ローカル MTU が最小の場合が多い
- TCP は MSS(Maximum Segment Size)を交換
- 1500 が大多数

経路上で MTU が小さくなるのはせいぜい 1 回
- 経路 MTU は一方向

経路 MTU 探索
- ローカル MTU で送信
- ルータからのパケット過大メッセージにより補正



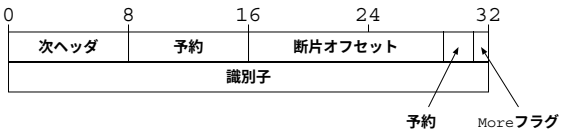
分割&再構成

分割は回避した方がよい
- TCP の通信単位(セグメント)にあわせる
- UDP や IP では分割は避けられない

ルータでは分割しない
- パケットは破棄
- パケット過大メッセージを返す

始点ホストのみで分割

断片ヘッダ



経路制御

IGP (Interior Gateway Protocol)

- RIPng
  IPv6 のみ、最大ホップ数は 15 のまま
- OSPF
  ルータ識別子とアドレスの分離

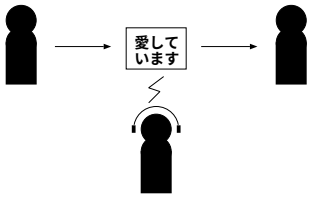
EGP (Exterior Gateway Protocol)

- ○ BGP4+
- × BGP5
- × IDRP

セキュリティに求められる機能

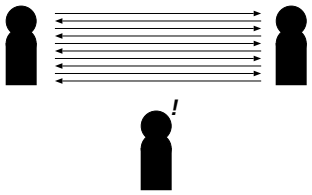
機密性
秘匿性
完全性
認証
否認防止
いやがらせに対する防御

機密性



通信当事者だけが内容を理解できること
第三者が盗聴できないこと
暗号で保護する
- 共有鍵暗号
- 公開鍵暗号

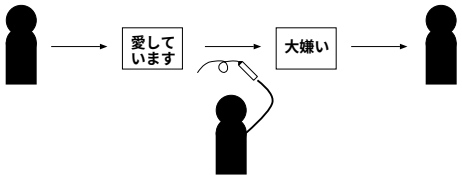
秘匿性



通信当事者だけが通信の発生を認識できること
第三者に通信の発生を知られないこと
- 内容が分からなくても有益な情報を得られる
- ラフィック解析

インターネットでは秘匿性を保護するのは困難

完全性



送信者の意図する内容がそのまま受信者に届けられること
第三者の改竄を検知できること
電子署名により保護できる
- ハッシュ関数、公開鍵暗号

HMAC により保護できる
- ハッシュ関数、秘密の共有

認証

ある名前を名乗る人が本当にその人だと確認すること
なりすましを防止すること
その人しか知らない秘密を確認する
- あらかじめ登録した秘密
- 電子署名

否認防止 & いやがらせに対する防御

通信内容を後から否定できないこと
- 電子商取引には必須
- 適切な公開鍵暗号で実現可能

いやがらせの例
- 多量のパケットを送り付ける

工夫はできるが...
- Fair queue など

完全な防御方法はない

IPsec の必要性

セキュリティの機能
- 機密性、完全性、認証、否認防止

他の層にもセキュリティ機能がある
- ソッケト層: SSL (Secure Socket Layer)
- アプリケーション層: SSH (Secure Shell)
- アプリケーション層: PGP/MIME、S/MIME

ネットワーク層にセキュリティ機能は必要か
- ネットワーク層で認証できないと保護できない攻撃がある
- (例)TCP リセット攻撃
- 抽象度が高い

IPsec は必要である

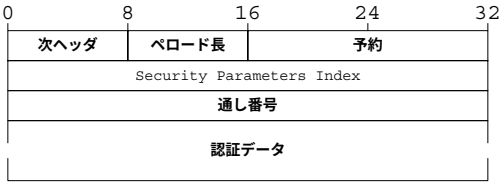
IPsec

認証ヘッダ(AH: Authentication header)
- 完全性、認証、否認防止
- 暗号の輸出規制のため暗号ペイロードから切り離された
- 今や IP ヘッダも保護するときのみに利用

暗号ペイロード
(ESP: Encapsulating Security Payload)
- 機密性、完全性、認証、否認防止

変換 (Transform)
- 認証ヘッダや暗号ペイロードは枠組のみ提供
- 具体的な方式は変換で定める

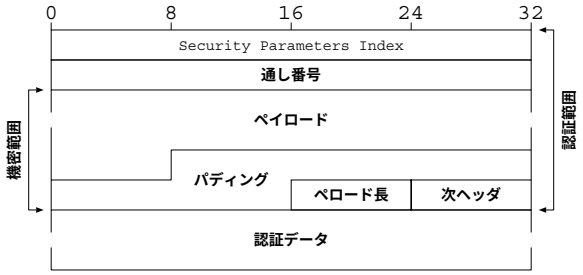
認証ヘッダ



SPI
- Security Parameters Index
- 方式は隠蔽されている

通し番号によるリプレイ攻撃の防止
- RFC 1826 には無かった...

暗号ペイロード



通し番号によるリプレイ攻撃の防止

- RFC 1827 には無かった...

IPsec のモード

トランスポート・モード

- 始点ホストが AH や ESP を作る

IPv6	AH	TCP

IPv6	ESP	TCP

トンネル・モード

- トンネル・ルータが AH や ESP を作る
- VPN (Virtual Private Network)

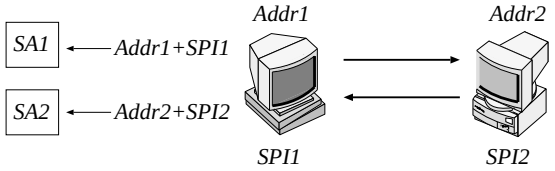
IPv6	AH	IPv6	TCP

IPv6	ESP	IPv6	TCP

VPN

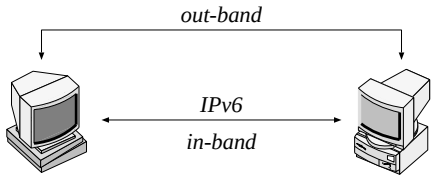
鋼鉄のパイプ
- 専用線で各部署を接続するのは高価
- サイトの出口で暗号トンネルを張る
- 安価なインターネット上にプライベート・ネットワークを構築

方式の合意



SA(Security Association)
- 始点アドレスと SPI から検索 (一方向)
- AH or ESP？
- 公開鍵暗号
- 共有鍵暗号
- セッション鍵
- ハッシュ関数

鍵配送



帯域内(in-band)
- SKIP

帯域外(out-band)
- Photuris、ISAKMP/Oakley

標準
- ISAKMP/Oakley は IPv6 で必須、IPv4 でオプション
- SKIP は IPv4、IPv6 ともにオプション

暗号の策定

多くのハッシュ関数はフリー
- MD5、SHA-1、RIPE MD160

Encumbered でない暗号が必要
- 特定の企業の特許にじゃまされない

公開鍵暗号
- RSA → Diffie-Hellman

共有鍵暗号
- DES、IDEA → CAST

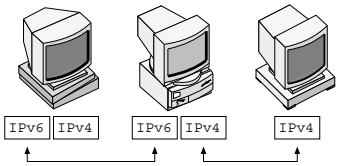
移行

IPv4 が多く IPv6 が少ない時期
- デュアル・スタック、トンネル

IPv4 と IPv6 が混在する時期

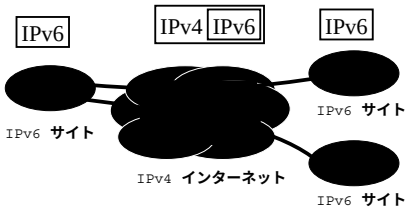
IPv6 が多数派、IPv4 が小数派になる時期
- トランスレータ

デュアル・スタック



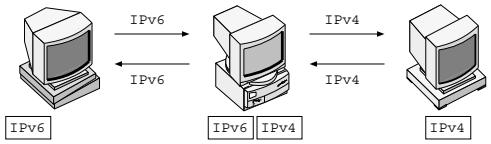
デュアル・スタック・ホスト
- 初期の IPv6 ホストは IPv4 もしゃべる必要がある
- IPv4 ホストとは IPv4 で
- デュアル・スタック・ホストとはどちらでも可
- DNS は IPv4 で検索

IPv6 over IPv4 トンネル



IPv6 サイトは IPv4 インターネットに浮かぶ島
IPv6 島を接続
- IPv4 インターネットをデータリンクだと思う
- IPv6 パケットを IPv4 パケットにカプセル化

トランスレータ



新たな IPv4 ホストは出現しなくなる
- IPv4 アドレスはいずれ枯渇する
- 2 つのスタックを管理するのはコストが高い

しかし、IPv4 ホストは残る
通訳が必要

6bone の歴史と現状

WIDE 6bone
初期の world-wide 6bone
現在の world-wide 6bone

WIDE 6bone

1996年6月9日
- 東京 NOC と奈良 NOC を結ぶ

静的な経路制御の破綻
RIPng による経路制御
特徴
- さまざまなデータリンク
  Eternet、FDDI、高速シリアル、ATM、トンネル....
- 多数のベンダー
  アカデミック、日立、東芝、富士通、NEC、Bay Networks...

ホームページ
- http://www.v6.wide.ad.jp/

初期の world-wide 6bone

1996年3月
- IETF@Los Angeles で村井、Postel、Deering が会合

1996年6月
- IETF@Montoreal で第1回 6bone BOF
- 1996年7月15日に world-wide 6bone を始めると決定
- WIDE、UNI-C、G6 が参加を表明

約束の日
- UNI-C、G6 共にダメ
- 7月16日に WIDE は Cisco とトンネルを張る
- 7月18日に G6 とも張る

RIPng による経路制御

ホームページ
- http://www.6bone.net/

現在の world-wide 6bone

RIPng の破綻

経路集約型アドレスの出現

リナンバーリング
- 1997年10月1日〜11月1日
- 経路集約型アドレスへの移行
- RIPng から BGP4+ への移行

41 個の pTLA
- WIDE は 5