＜添付2＞

Asterisk の IPv6 対応について

2012年12月3日公開
IPv6普及・高度化推進協議会
IPv4/IPv6共存WG
アプリケーションのIPv6対応検討SWG

Asteriskは米国Digium社の登録商標または商標です。
そのほかの記載の会社名、製品名は、それぞれの会社の
商標もしくは登録商標です。

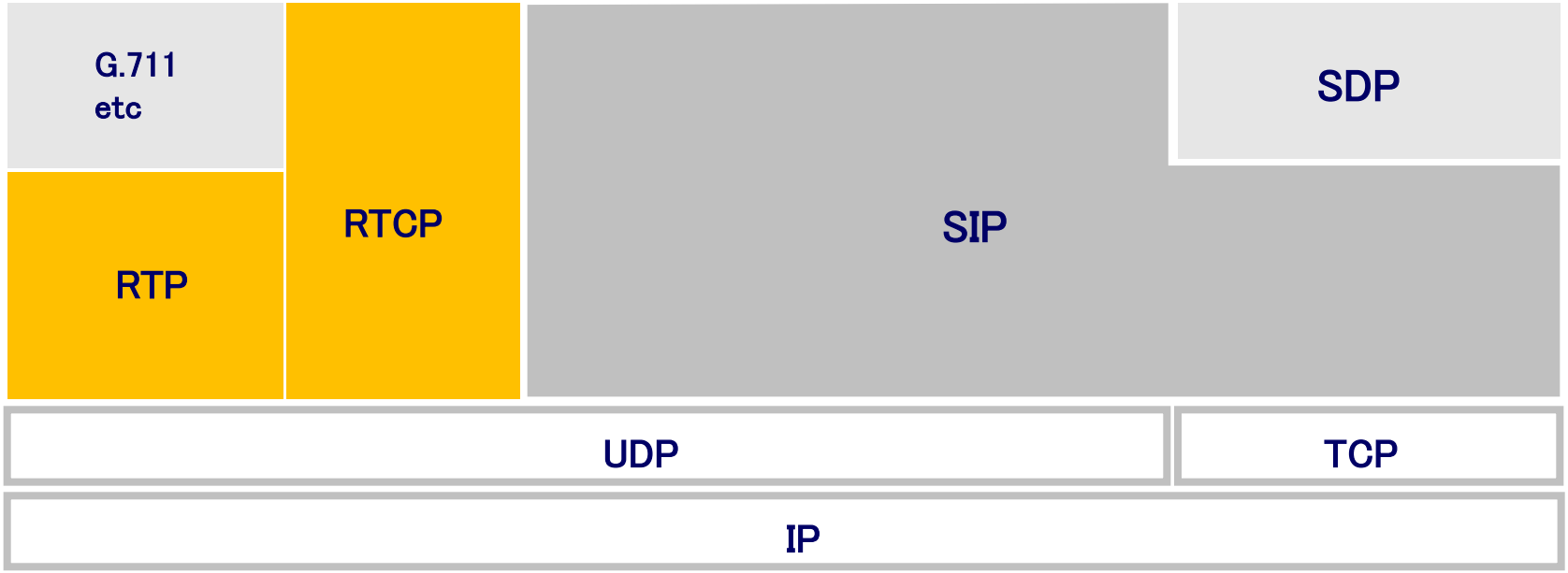
はじめに

- IPv6に対応したアプリケーション開発に必要な要素として、開発環境、ツール、そしてコーディングなどがあげられるが、これらに関し、必要な情報が不足している
- 本SWGでは、必要な情報の整理・検討を実施し、SocketプログラミングとWebサービスに大別したうえでそれぞれのガイドラインを作成することにした
- Socketプログラミングに関する検討では、アプリケーションの例としてAsterisk(IP-PBXソフトウェア) を対象にIPv6対応方法に関する調査をした
- 本書では、AsteriskのIPv6対応方法の調査結果を述べる。Asteriskでは、SIPを用いているため、SIPについても解説する

SIP

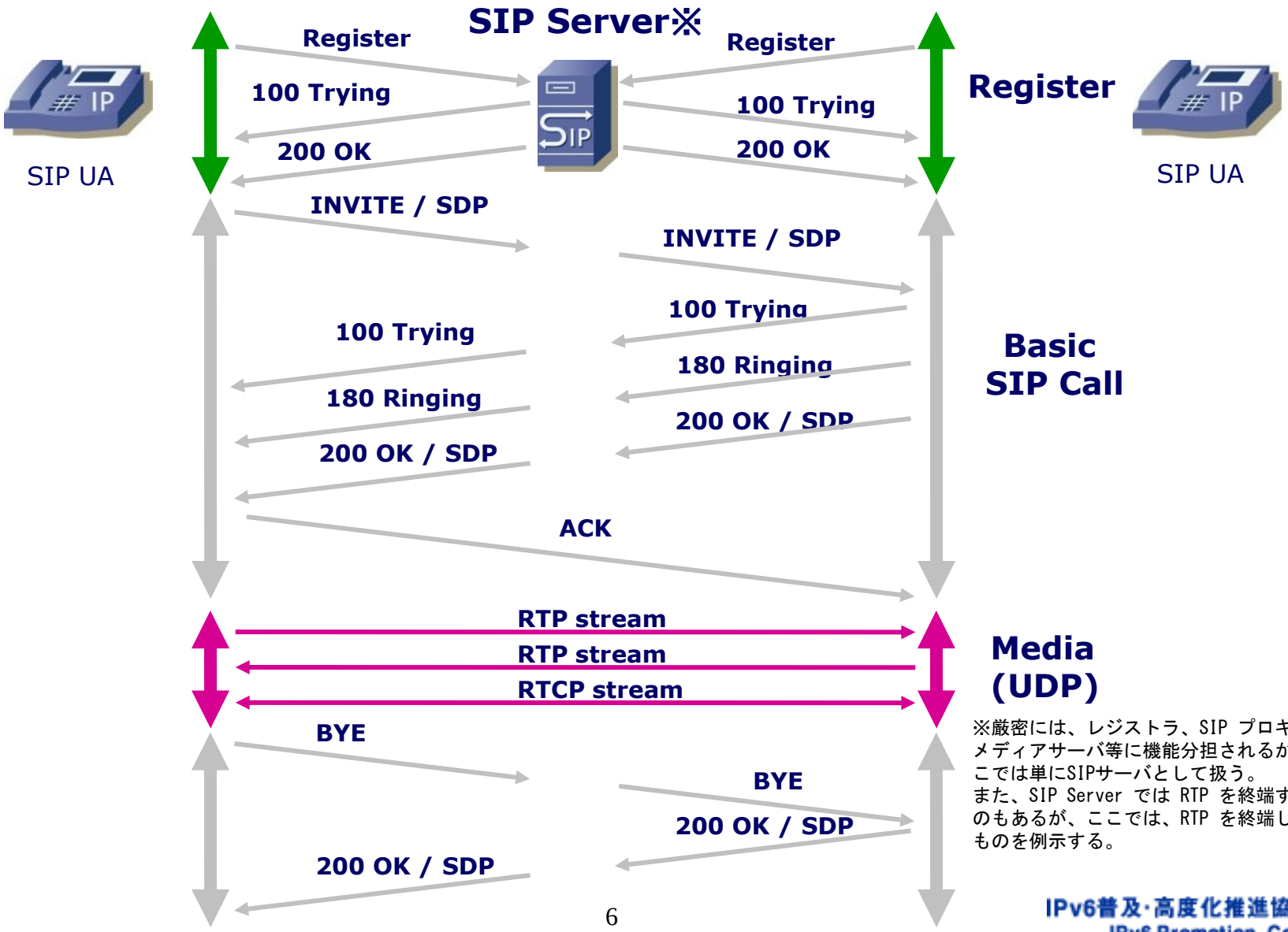
- SIP(Session Initiation Protocol)とは、2つ以上のクライアント間でセッションを確立するためのIETF標準の通信プロトコル
- IETFにて、汎用のセッション制御プロトコルとして開発された
- RFC3261：Session Initiation Protocol
- RFC4566：Session Description Protocol
- 特徴として、HTTPに似た、テキストベースのリクエストとレスポンスによって通信を行い、相手先(通話先) はURI(Uniform Resource Identifier)を指定
- URIで得られる通信先は、下位レイヤではIPの通信先として処理
- SIPサーバは次の機能から構成
  - プロキシサーバ
  - リダイレクトサーバ
  - ロケーションサーバ
  - レジスターサーバ

SIP Protocol Architecture



G.711: ITU-T standard for audio companding.
SDP: Session Descripton Protocol
RTCP: RTP Control Protocol
SIP: Session Initiation Protocol
RTP: Realtime Transport Protocol
TCP: Transmisson Control Protocol
UDP: User Datagram Protocol
IP: Internet Protocol

SIP の一般的な接続シーケンス



※厳密には、レジストラ、SIP プロキシ、メディアサーバ等に機能分担されるが、ここでは単にSIPサーバとして扱う。
また、SIP Server では RTP を終端するものもあるが、ここでは、RTP を終端しないものを例示する。

Asterisk について

- オープンソースの IP-PBX ソフトウェア
（IPネットワーク内で、IP電話端末の回線交換を行う装置およびソフトウェア）
- http://www.asterisk.org/
- 複数のバージョン（1.4.x、1.8.x、10.x、11.x）
- 呼制御
- SIP/IAX2/H.323 などに対応
- メディア
- 音声：G.711、G.722、G.729等に対応
- 映像：H.263、H.264 等に対応（トランスコードは不可）

Asterisk の IPv6 対応について

- バージョン 1.8 系より対応
  （最新版は、11.0.1（2012/11/19現在））
- IPv6 対応箇所
- 呼制御（SIP/IAX）
  SIP は、UDP/TCP/TLS に対応
- 管理機能（設定用 Web インタフェース、AMI:Asterisk Management Interface）
- メディアトランスポート（RTP/SRTP）
- IPv4/IPv6 の相互接続について
- B2BUA（ゲートウェイ）を用いた接続形式で、IPv4 端末と IPv6 端末との相互接続が可能（この場合、Asterisk が動作している計算機の OS がデュアルスタックで動作していることが前提）

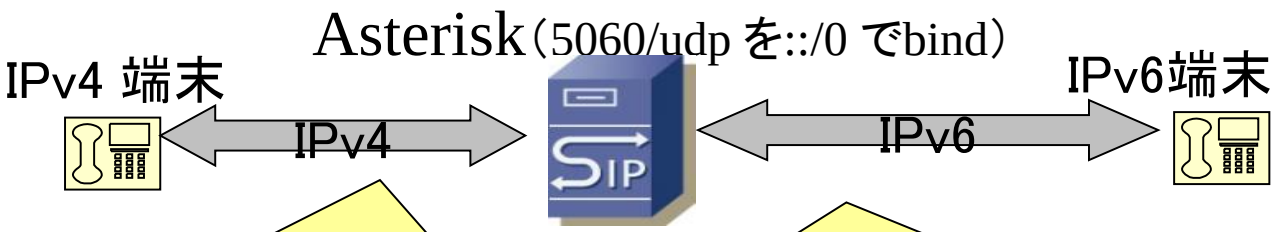
Asterisk の主要機能と IPv6対応

大項目	中項目	概要	IPv6 対応
呼制御	SIP	SIP による呼制御機能	○
	IAX	IAX による呼制御機能	○
	H.323	H.323 による呼制御機能	×
	Websocket	Websocket による呼制御連携	○
メディア処理	RTP	音声/映像ストリーム	○
暗号化	SIPS 対応	SIP over TLS	○
	SRTP	暗号化 RTP	○
管理機能	AMI	Asterisk Management Interface	○
	Web インタフェース	ブラウザからの設定機能	○
PBX 間連携	DUNDi	Asterisk 間の相互接続機能	×

2012/11/1 現在（Asterisk 11）

接続例（Asterisk が B2BUA で動作）

- SIP メッセージ例
  - 転送相手がIPv4かIPv6を、Asterisk への登録情報を元に判定し、それぞれのプロトコルに応じて SIP メッセージを組立て、転送する




```
INVITE sip:6001@192.168.1.212:51784;rinstance=51c2ff3360b4f8d6;transport=udp SIP/2.0
Via: SIP/2.0/UDP 192.168.1.214:5060;branch=z9hG4bK727d226d
Max-Forwards: 70
From: "ekiga" <sip:6000@192.168.1.214>;tag=as3350c74c
To: <sip:6001@192.168.1.212:51784;rinstance=51c2ff3360b4f8d6;transport=udp>
Contact: <sip:6000@192.168.1.214:5060>
Call-ID: 4c9d4cb6326cf955529abb9b7a253f6d@192.168.1.214:5060
CSeq: 102 INVITE
User-Agent: Asterisk PBX 10.0.0-rc1
Date: Wed, 16 Nov 2011 13:50:02 GMT
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY,
INFO,PUBLISH
Supported: replaces, timer
Content-Type: application/sdp
Content-Length: 442

v=0
o=root 318336445 318336445 IN IP4 192.168.1.214
s=Asterisk PBX 10.0.0-rc1
c=IN IP4 192.168.1.214
b=CT:384
:
```

SIP メッセージ中のアドレスについて、IPv4/IPv6 のチェックを行っている

```
INVITE sip:6001@ietugu.example.com SIP/2.0
Via: SIP/2.0/UDP [2001:db8:1::215:c5ff:fe12:2c68]:5060;branch=z9hG4bK1900520024
From: <sip:6000@ietugu.example.com>;tag=623328519
To: <sip:6001@ietugu.example.com>
Call-ID: 134945149
CSeq: 20 INVITE
Contact: <sip:6000@[2001:db8:1:1:215:c5ff:fe12:2c68]>
Content-Type: application/sdp
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE,
SUBSCRIBE, INFO
Max-Forwards: 70
Subject: Phone call
Content-Length: 384

v=0
o=6000 123456 654321 IN IP6 2001:db8:1:1:215:c5ff:fe12:2c68
s=A conversation
c=IN IP6 2001:db8:1:1:215:c5ff:fe12:2c68
b=AS:1024
t=0 0
m=audio 7078 RTP/AVP 0 101
:
```

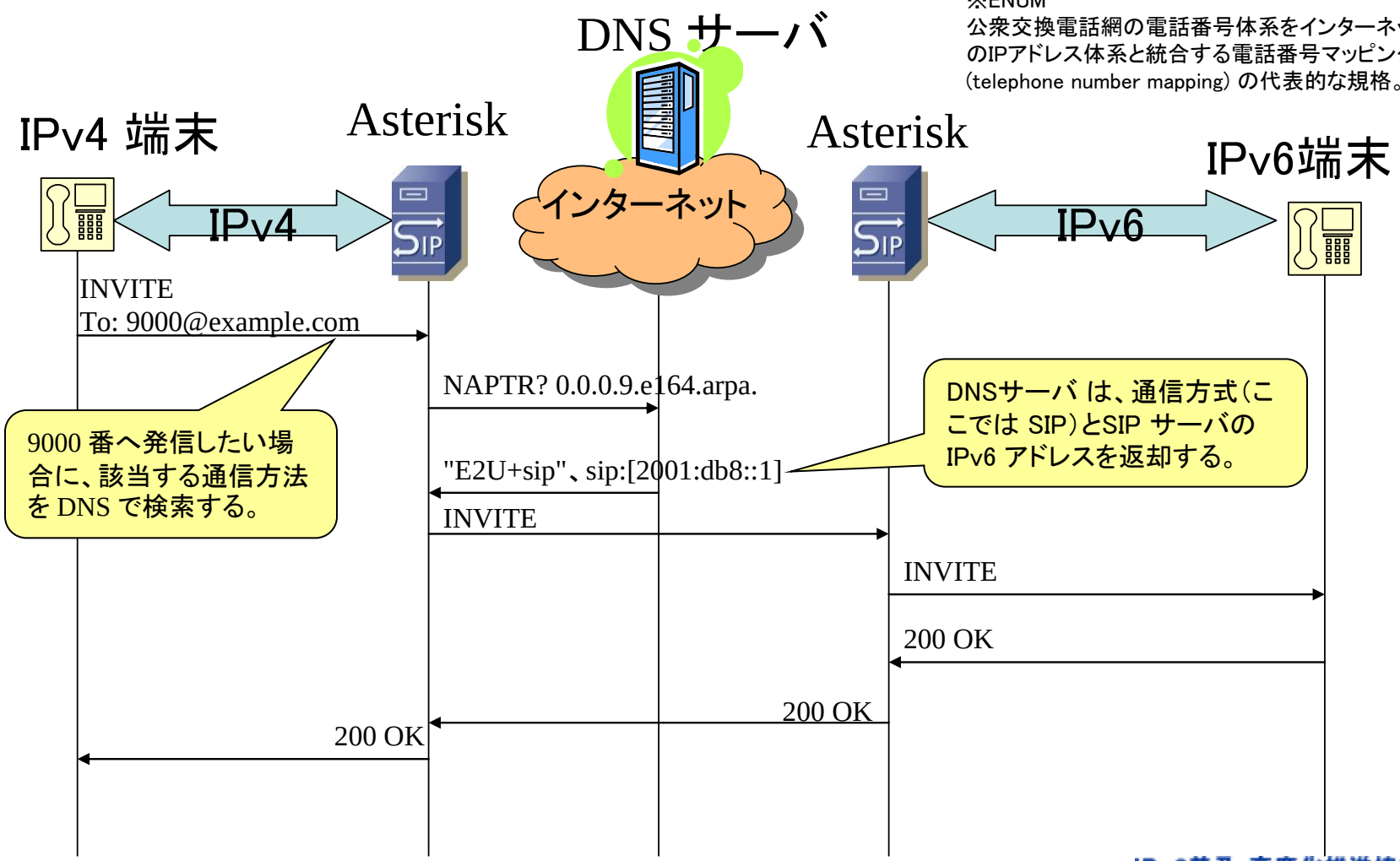
接続情報を、転送先にあわせてIPv6 アドレスを使用して作成する

10

接続例2（ENUMによる名前解決）

- Asterisk は ENUM※にも対応しており、インターネット上の VoIP 接続も可能。
- 名前解決は標準ライブラリ関数の *res_search()* を使用するため、IPv6 対応はOS のリゾルバライブラリに依存する。

※ENUM
公衆交換電話網の電話番号体系をインターネットのIPアドレス体系と統合する電話番号マッピング（telephone number mapping）の代表的な規格。



設定ファイル

- bind() を行うアドレスは IPv4/IPv6 の両方が記述可能（設定ファイルは sip.conf）
- IPv4 example: bindaddr=0.0.0.0:5062
  bindaddr=0.0.0.0
- IPv6 example: bindaddr=[::]:5062
  bindaddr=::
- bindaddr=0.0.0.0 とした場合、IPv4 のみ受信可能
- bindaddr=<特定の IPv4/IPv6> アドレスを指定した場合は、そのアドレスファミリのみ受信可能
- デュアルスタックにする場合は、bindaddr=:: とするが、挙動はOS依存である
- IPV6_V6ONLY マクロを意識した設定項目はない
- Linux の場合、/proc/sys/net/ipv6/bindv6only=0 としないとデュアルスタックにはならない

ログファイル

- ログファイル中の IPv6 アドレス表示は '[<IPv6 アドレス>]:<ポート番号>' となる

例

```
[Nov 16 22:50:02] VERBOSE[6103] chan_sip.c: Peer
audio RTP is at port
[2001:db8:1:1:215:c5ff:fe12:2c68]:7078
[Nov 16 22:50:02] VERBOSE[6103] chan_sip.c: Peer
video RTP is at port
[2001:db8:1:1:215:c5ff:fe12:2c68]:9078
```

ソケットまわり

- アプリ内でライブラリ化されており、getaddrinfo() などの関数は隠蔽されている

アドレス情報の保持（1）

- 従来は *struct sockaddr_in* を使用していたのが、*struct sockaddr_storage* を使用するように修正されている

asterisk 1.4.40 の場合

channel/chan_sip.c より

```
struct sip_pvt {
:
struct sockaddr_in sa;
};
struct sip_peer {
:
struct sockaddr_in addr;
};
```

一部のOS（MacOS 等）では構造体の長さが必要なため、移植性を考慮して追加されている

asterisk 10.3.1 の場合

include/asterisk/netsock2.h より

```
struct ast_sockaddr {
struct sockaddr_storage ss;
socklen_t len;
};
```

channel/sip/include/sip.h より

```
struct sip_pvt {
:
struct ast_sockaddr sa;
};
struct sip_peer {
:
struct ast_sockaddr addr;
};
```

アドレス情報の保持(2)

- アドレスの文字列を格納する文字配列の要素数を *INET_ADDRSTRLEN* から *INET6_ADDRSTRLEN* へ変更している

asterisk 1.4.40 の場合

channel/chan_sip.c より

```
static void realtime_update_peer(…)
{
:
char ipaddr[INET_ADDRSTRLEN]

};
```

asterisk 10.3.1 の場合

channel/sip/include/sip.h より

```
static void realtime_update_peer(…)
{
:
char ipaddr[INET6_ADDRSTRLEN]

};
```

IPv6 対応汎用関数(1)

main/netsock2.c:

```
int ast_sockaddr_split_hostport(char *str, char **host, char **port,
                                int flags)
```

IP アドレスとポートを分離する

ex) [2001:db8:cafe:babe::1]:5060 => 2001:db8:cafe:babe::1 と 5060 に分離

```
int ast_sockaddr_resolve(struct ast_sockaddr **addrs,
                         const char *str, int flags, int family)
```

アドレス解決を行う。引数にアドレスファミリーが入る

検索されたアドレスすべてを、引数で与えられた領域に保存する

この関数を利用して、channels/chan_sip.c で下記の関数が定義されている

```
static int ast_sockaddr_resolve_first_af(struct ast_sockaddr *addr,
                                         const char* name, int flag,
                                         int family)
```

指定したアドレスファミリで検索し、先頭の１つだけを返却する

```
static int ast_sockaddr_resolve_first(struct ast_sockaddr *addr,
                                      const char* name, int flag)
```

設定ファイルで指定されたbindaddrのアドレスファミリで検索し、先頭の１つだけ返却する

IPv6 対応汎用関数(2)

```
int ast_sockaddr_ipv4_mapped(const struct ast_sockaddr *addr, struct
                                ast_sockaddr *ast_mapped)
```

IPv4 mapped address（実体は *struct sockaddr_in6*）が格納された ast_sockaddr を、IPv4 アドレスが格納された形式（実体は struct sockaddr_in）に変換する

```
int ast_set_qos(int sockfd, int tos, int cos, const char *desc)
```

IP ヘッダの TOS フィールド、もしくは IPv6 ヘッダの Traffic Class フィールドを設定する（Linux の場合は、*SO_PRIORITY* ソケットオプションにより、cos の値が設定される）

接続

- 設定ファイルに書かれている IP アドレスを *struct in_addr* もしくは *struct in6_addr* に変換するために *getaddrinfo()* を使用している
- *getaddrinfo()* は、前述の *ast_sockaddr_resolve()* 経由で呼び出されており、場合によっては、取得されるアドレスリストの先頭のみを使用している

まとめ

- アドレス操作関数は、IPv4/IPv6 を意識しているものの、局所化できるように内部関数を作成して使用している

参考文献(Webサイト)

- SIP IPv6 関連
  - SIP FORUM IPv6 task group
    http://www.sipforum.org/content/view/398/286/
- Asterisk 関連
  - Asterisk wiki
    http://wiki.asterisk.org
- IPv6 関連
  - Internet Week 2011より 事例から学ぶIPv6トラブルシューティング
  - http://www.nic.ad.jp/ja/materials/iw/2011/proceedings/t2/t2-02.pdf