Porting Asterisk to IPv6

Marc Blanchet

Frédérick Lefebvre

Viagénie

http://www.viagenie.ca

Presented at Astricon, Dallas, October 2006

Plan

- IPv6
- Why IPv6 and Asterisk
- New API
- IPv6 in chan_sip
- New ast_vinetsock API
- New functions
- Sip.conf
- Demo
- Next Steps
- Conclusion

Why US?

- Viagénie: consulting and R&D in networking.
- Started to work on IPv6 in 1996.
- Contributions to 6bone, IETF; invented/operated: 6tap, freenet6.net; ported Quake[1,2] to IPv6; wrote RFCs; co-founded IPv6Forum, member of North American IPv6 Task Force. Founded Hexago.
- Wrote Cisco IPv6 course; authored the “Migrating to IPv6” book published by Wiley (2006).
- Helping enterprise, gov, providers to deploy IPv6. Helping applications developers to port to IPv6.
- Our goal: infect Asterisk with the IPv6 virus! ;-)

IPv6?

- New version of IP:
  - fixes IPv4 issues
  - adds functionality
- Addresses:
  - 128 bits
  - written in hex with ':' as separator; method to compress the writing: all zeros = '::'
  - 2001:db8:1:1::1
  - In URL: enclose with []: 'http://[2001:db8:1:1::1]:8080'
  - Loopback: '::1'
  - Link(Subnet,vlan,...) mask is fixed: /64
  - Unique private address space: no collision of private networks

IPv6?

- Addresses (cont):
  - Scoped addressing: link scope, site scope. An enabled IPv6 stack has already an IPv6 address (link scope) on each interface, even if no IPv6 external connectivity.
  - Multiple addresses per interface: link-scope, global, [site,...]
  - No NAT.
- Mobility: keep connections up even when host changes IP address
- Autoconfiguration: routers announce the link prefix on the link. Hosts use their MAC address for the host part of the address
- Integrated IPsec
- Many more features

IPv6 Market

- IPv4 address depletion: remaining: <25% of usable address space. Using ~5%/year. Predictions of exhaustion for 2009-2011.
- Asia
  - Japan: see http://www.v6pc.jp
  - China: through NGN. Olympics is important milestone.
- US government:
  - Mandating IPv6 for 2008 in all agencies
  - DoD is leading
- Providers (short list):
  - Teleglobe/VSNL, NTT, AT&T, GlobalCrossing,...
  - Comcast: can't address all their devices (100M+) with IPv4. Deploying IPv6. (DOCSIS 3.0 is IPv6-ready).

IPv6 Support

- Support on OS (stack and API):
  - Same (new) API everywhere!!! ;-)
  - Since: Linux 2.4, FreeBSD 4.X, MacOSX 10.2, Windows XP, Solaris 8, ...
- Opensource Apps: Apache 2.0+ (1.3 with a patch), Sendmail/postfix, openssh, Xfree/Xorg, ...
  - Now Asterisk... ;-)
- Support on network gear: Cisco, Juniper, Checkpoint, Quagga/Zebra, ...

Why IPv6 and Asterisk?

- IPv6 and SIP
  - delivers direct end-2-end reachability between any host.
  - No NAT, No STUN, No TURN, No ICE, No MIDCOM, = no complexity, “just works”.
  - True end-2-end media path.
  - Much easier to deploy. A VoIP-IPv6 deployment in Japan found important cost reductions because of the ease of installation and support.
- To have an IPv6-enabled application such as Asterisk, need to convert to the new API.

New API

- New API for IPv6 [RFC3493, RFC3542]
  - Makes the application version independent. The stack chooses which IP version will be used for that connection.
  - A ported application becomes IP version unaware.
  - No change to socket(), bind(), listen(), accept(), connect(), recv(), send(), close()...
- Changes:
  - Struct hostent replaced by struct addrinfo
    - Addrinfo is a linked list of addresses
    - It contains everything needed to initialize a socket.

New API

- Changes:
  - sockaddr record
    - sockaddr_in : IPv4
    - sockaddr_in6 : IPv6 only. Do not use.
    - sockaddr_storage: version independent for memory allocations.
    - sockaddr *: for casting
  - gethostbyname replaced by getaddrinfo
  - gethostbyaddr, inet_addr, inet_ntoa replaced by getnameinfo
- More considerations:
  - Parsing URLs: need to take care of the IPv6 syntax (i.e. [])
  - Parsing and storing IP addresses

New API

- History:
  - New API had multiple revisions, based on feedback of porting, deployment and engineering.
  - Documentation and “old” code still uses old api calls.
  - Old ways:
    - IPv4-mapped addresses: important security issues.
    - Old calls (deprecated, nowadays no more available in some OS):
      - gethostbyname2()
      - getipnodebyname()
      - getipnodebyaddr()

Best Practices for API usage

- Use sockaddr_storage for storing sockaddrs.
- Use sockaddr * for pointer to sockaddrs
- Always pass and carry the sockaddr length to be fully portable across OS platforms.
- After the getaddrinfo() call, go through the link list of addrinfo to connect.
- Parse addresses and URL to support both IPv4 and IPv6 addresses (with port numbers) syntax.
- Do not use IPv4-mapped addresses, old API calls (gethostbyname2(), getipnode*())

Challenges with IPv6 in Asterisk chan_sip

- Current architecture supports a single socket : 'sipsock'.
- The default source address is hardcoded to 0.0.0.0.
- The RTP socket is initialized from 'sipsock'
- Widespread use of sockaddr_in structures and short buffers (>256 bytes) to store hostnames and IP address strings.
- Many instances of similar code for parsing SIP url.

Design choices

- Use multiple sockets
  - Initial patch provides 1 socket per address family.
  - future work should include multiple sockets for each address family.
- Version independent when possible
  - Whenever possible, do not use sockaddr_in or sockaddr_in6 and never guess at the length of a sockaddr structure.
  - Only exception should be for setting socket options.

Code changes

- Replaced all use of sockaddr_in in data structures with sockaddr_storage.
- Associates a socklen_t element with each sockaddr_storage.
  - the socklen member is only initialized when a sockaddr_in of sockaddr_in6 structure is copied in the allocated memory... never when the memory is allocated.
- Use the new ast_vinetsock API

New ast_vinetsock API

- ast_netsock (netsock.h) is currently used in chan_iax, not in chan_sip.
- ast_netsock has link lists to manage multiple sockets.
- the ast_netsock API was augmented to support IPv6.
- New and modified functions are in the new ast_vinetsock namespace (defined in netsock.c): no collision with ast_netsock.
- 3 types of functions are defined in ast_vinetsock:
  - Address string parsing.
  - Address structure handling.
  - Socket management.

String parsing functions

- Parse host:port and address strings in a version independent way.
- Used for:
  - Parsing and validation of configuration files.
  - Parsing SIP header fields such as 'contact' and 'via'.
-
- Db store uses ':' between fields. ':' is used in IPv6 address. Enclosing IPv6 address in []. Impact for other db readers.

Address structure handling functions

- Initialize sockaddr structures from strings.
- Extract data from sockaddr structures.
- Build host:port and address strings from sockaddr structures.
- Used for:
  - Selecting a source address.
  - Printing addresses and host:port strings to logs and console.
  - Building SIP/SDP fields from address structures.

Socket management functions

- Initialize sockets through ast_vinetsock structures.
- Set socket options.
- Bind on sockets and register callback functions.
- Used for:
  - Initializing IP listeners

Modifications to sip.conf

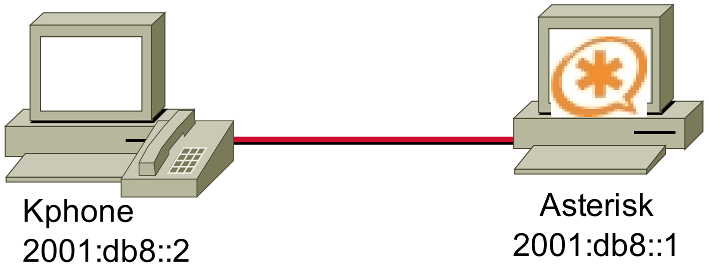
- 'bindaddr' now supports the address:port syntax such as:
  - 10.1.1.1
  - 10.1.1.1:5060
  - [2001:db8::1]
  - [2001:db8::1]:5060
- If no 'bindaddr' is specified for an address family, the wildcard is used (0.0.0.0 and [::]).
- 'host' contains only the address, therefore no brackets.
- 'bindport' is still supported for backward compatibility.

IPv6 SIP user agents

- Few open source IPv6 SIP user agents are available at this time.
- Makes testing and especially interop testing more limited.
- We tested 2 softphones with a 'working' ipv6 implementation:
  - kphone 3.1.1 with IPv6 patch.
  - Linphone 1.3.5
  - Both implementations contains (IPv6) bugs.

'Hello World' demo

- Uses Kphone as IPv6 SIP UA.
- Register to Asterisk.
- Make a call to play the 'Hello world' sound file.

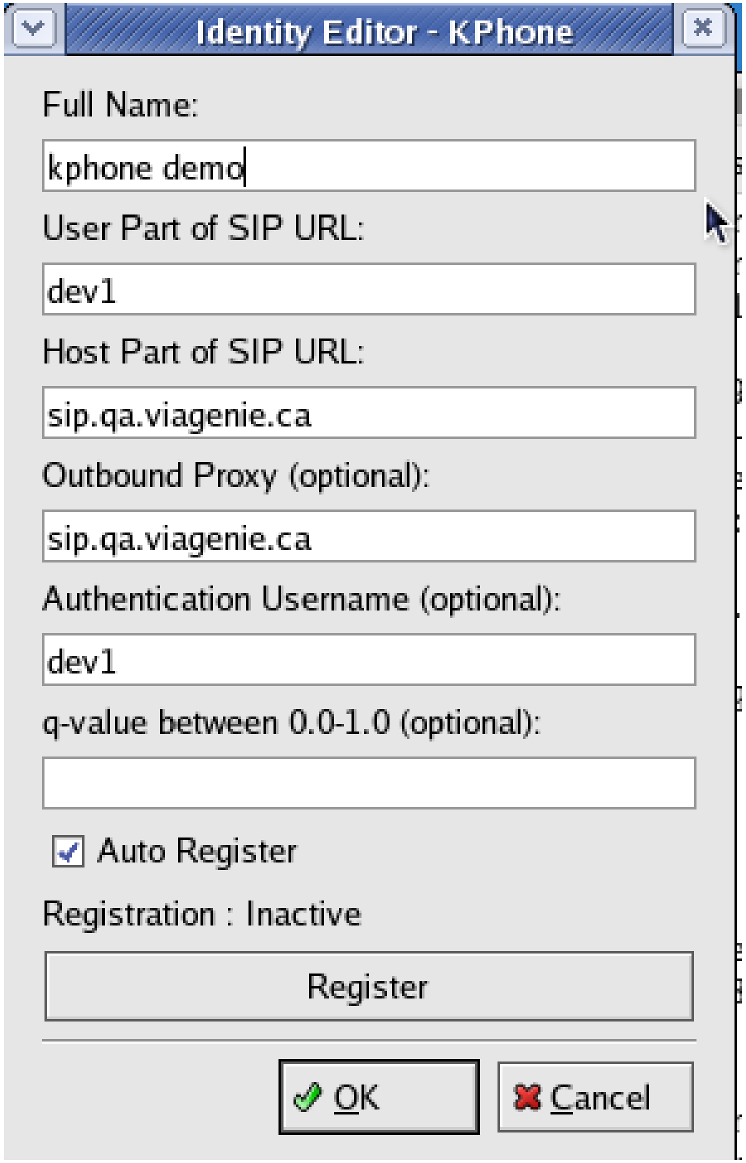


'Hello World' demo (cont.)

```
[general]
context=internal
bindaddr=[2001:db8::1]
allow=ulaw

[dev1]
type=friend
host=dynamic
context=internal
disallow=all
allow=ulaw

[dev2]
type=friend
host=dynamic
context=internal
disallow=all
allow=ulaw
```



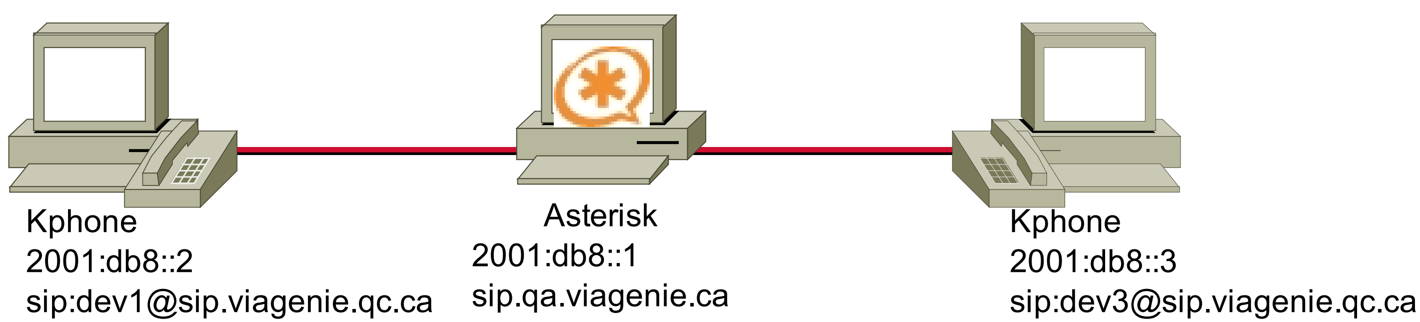
'Hello World' demo (cont.)

```
Reliably Transmitting (no NAT) to [2001:db8::2]:5060:
SIP/2.0 200 OK
Via: SIP/2.0/UDP [2001:db8::2];received=2001:db8::2
From: "Fred" <sip:dev1@sip.qa.viagenie.ca>;tag=61617230
To: <sip:2@sip.qa.viagenie.ca>;tag=as15d09daf
Call-ID: 336600123
CSeq: 3245 INVITE
User-Agent: Asterisk PBX
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY
Supported: replaces
Contact: <sip:2@[2001:db8::1]>
Content-Type: application/sdp
Content-Length: 168

v=0
o=root 1406 1406 IN IP6 2001:db8::1
s=session
c=IN IP6 2001:db8::1
t=0 0
m=audio 10610 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=silenceSupp:off - - - -
a=sendrecv
```

Bidirectional call demo

- 2 Kphone IPv6 SIP User Agents register to an Asterisk server.
- Establish a SIP call between the two user agents through an extension on Asterisk.



Bidirection call demo (cont.)

```
Reliably Transmitting (no NAT) to [2001:db8::3]:5060:
INVITE sip:lefebvre@[2001:db8::3];transport=udp SIP/2.0
Via: SIP/2.0/UDP [2001:db8::1]:5060;branch=z9hG4bK1dc90af0;rport
From: "Fred" <sip:dev1@[2001:db8::1]>;tag=as3038e677
To: <sip:lefebvre@[2001:db8::3];transport=udp>
Contact: <sip:dev1@[2001:db8::1]>
Call-ID: 5351c608290f3c9d03ab0e346ed44a80@2001:db8::1
CSeq: 102 INVITE
User-Agent: Asterisk PBX
Max-Forwards: 70
Date: Wed, 18 Oct 2006 19:38:06 GMT
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY
Supported: replaces
Content-Type: application/sdp
Content-Length: 224

v=0
o=root 1406 1406 IN IP6 2001:db8::2
s=session
c=IN IP6 2001:db8::2
t=0 0
m=audio 32770 RTP/AVP 0 101
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=silenceSupp:off - - - -
a=sendrecv
```

Impacts

- Files touched:
  - netsock.c/.h
  - chan_sip.c
  - rtp.c
  - Few others
- Some numbers:
  - ~25% of functions were changed/touched
  - ~ thousand lines changed/touched.
  - “Everywhere” in chan_sip, because: networking, logging (printing addresses) and sip url parsing.

Next Steps

- Code is based on august 2006 trunk. Need to remerge to 1.4 and trunk.
- Running in production in our office and remote sites with IPv6 **and** IPv4 phones.
- Discuss with community how to integrate code into trunk (ongoing)
- Add a startup flag to Asterisk to disable IPv6.
- More testing! Especially Interop tests.
  - test with other implementations (SER, ...)
  - test with other IPv6 SIP UAs.... if you have one, please contact us.
- Improve IPv6 support in chan_sip to better handle complex scenarios.
  - implement ANAT [RFC4091, RFC4092].
  - IPv6 <-> IPv4
- Add IPv6 support to chan_iax (work in progress) and chan_*.
- Fix bugs

Conclusion

- Discussed:
  - the benefits of IPv6 and Why Asterisk benefits of being IPv6-enabled.
  - How to port an application to IPv6
  - Changes to Asterisk
  - Demo
  - Next Steps
- Information on this Asterisk-IPv6 project is available at:
  - http://www.asteriskv6.org .
  - We will be posting progress, tests with IPv6 UA, code, ...

Questions?

Contact info:

Marc.Blanchet@viagenie.ca

Frederick.Lefebvre@viagenie.ca

This presentation is available at http://www.viagenie.ca/publications/

Information on this Asterisk-IPv6 project: http://www.asteriskv6.org

References

– [RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, February 2003.
– [RFC3542] Stevens, W., Thomas, M., Nordmark, E., and T. Jinmei, "Advanced Sockets Application Program Interface (API) for IPv6", RFC 3542, May 2003.
– IPv6 Network Programming, Junichiro itojun Hagino, Elsevier, 2004, ISBN 1555583180.
– Migrating to IPv6, Marc Blanchet, Wiley, 2006, ISBN 0-471-49892-0, http://www.ipv6book.ca