Aculab digital network access cards

Adding and Using IPv6 Capabilities Guide

Revision 1.0.3

PROPRIETARY INFORMATION

Aculab Plc makes every effort to ensure that the information in this document is correct at the time of publication. Aculab does not, however, accept responsibility for any errors or omissions. Aculab has a policy of continuous product introduction and improvement. Please check documentation and product versions to ensure accuracy of information.

Document Revision

Rev	Date	By	Detail
1.0.0	19.02.15	PGD	Initial release
1.0.1	24.02.15	EBJ	Updated to corporate fonts and reformatted.
1.0.2	11.03.15	PGD	Add ProsodyS note.
1.0.3	06.08.18	PGD	Add additional ProsodyS notes.

CONTENTS

1 Introduction

This document is a guide to adding IPv6 capabilities to applications using Aculab APIs for ProsodyS, ProsodyX and SIP related products. This is functionality introduced with the 6.6 Aculab Telephony Software distribution. Applications may be enhanced to allow IPv6 addresses to be specified in addition to those for IPv4, make and receive SIP calls over IPv6, transmit and receive RTP streamed audio using IPv6 packets, and control ProsodyX cards or ProsodyS servers over an IPv6 link. The main advantage to adding such capabilities is to allow systems to be deployed in environments where use of IPv6 networking is preferred (often allowing simpler network and address management), and to allow interaction with peer IPv6 enabled systems or devices.

This document assumes familiarity with equivalent IPv4 functionality in the TiNG and SIP APIs, and some familiarity with IP network terminology and address management issues. Full details of individual API calls mentioned here are given in respective API guides for those product elements.

It should be noted that Aculab IPv6 capabilities will only work on 6.6 and later supported O/S platforms, in particular the functionality described will not work on older versions of Windows.

2 IPv6 and VoIP

2.1 IPv6 Addresses, Local Links and Routers

Just a few concepts are briefly described here to fix the terminology used in the remaining part of the document.

An IPv6 address when represented as a text string is written as a series of colon separated hex words for example:

fdcd:fb9b:497a:0:34fc:2df9:2935:13cb

Groups of zero words can be abbreviated using adjacent colons, the two addresses below are equivalent:

fdcd:fb9b:0:0:0:0:0:13cb

fdcd:fb9b::13cb

Often, in order to avoid ambiguity with other syntax using a colon (for example port number suffix), the text version of the address is required to be surrounded by square brackets:

[fdcd:fb9b:497a:0:34fc:2df9:2935:13cb]

As with IPv4, the address space has a structure assigning certain ranges of addresses for certain types of use. A range of addresses can be specified with a prefix specification, for example to denote set of addressed whose first 16 bits are 0xFC44 this can be written:

fc44::/16

With respect to Prosody applications the following types of address are noteworthy:

- the loopback address - ::1 (like 127.0.0.1 for IPv4)
- link-local addresses - addresses prefixed with fe80::/10 (like 169.254.x.x for IPv4)
- global addresses - have meaning outside scope of local link

The local link is the name given to a segment of the IPv6 network where all attached devices (IPv6 endpoints) can communicate with each other without having to go through an intermediate router. This may be achieved using link-local addresses or a address with global prefix that is associated with the link (made known to endpoints on the link by a router attached to link).

A link-local IPv6 address may sometimes have a scope identifier suffix appended to it separated by a percent sign in order to indicate which local (NIC) interface it is associated with. On Windows it is generally a number while on Linux it is likely to be a device name, for example:

fe80::21c:23ff:fee2:85d%eth1

Normally an IPv6 endpoint will interact with an IPv6 router in order to be assigned an IPv6 address automatically. However for some configurations there may be a requirement to use statically assigned addresses, or at any rate addresses with a specific prefix rather than the one managed by a particular router. One reason this might be required is that automatically assigned IPv6 addresses reveal the MAC address of the equipment to remote parties which may in some circumstances not be desirable.

One convenient way of enabling a network for use with IPv6 and automatic address assignment is to set up a Linux/BSD box to act as an IPv6 router and run a suitably configured Router Advertisement Daemon (radvd).

2.2 IPv6 and SIP

In the SIP protocol, various IP addresses are conveyed in the exchanged data, in particular address information about caller, media destination addresses for RTP streams to be delivered to etc.

For an IPv6 SIP call, these would need to appear as IPv6 addresses in SIP protocol fields.

```
To: <sip:8986@[fd1b:81ba:ac00:25e7:c9ec:6d95:bbb1:83c0];user=phone>
SIP to address: sip:8986@[fd1b:81ba:ac00:25e7:c9ec:6d95:bbb1:83c0];user=phone
            SIP to address User Part: 8986
            SIP to address Host Part: [fd1b:81ba:ac00:25e7:c9ec:6d95:bbb1:83c0]
            SIP To URI parameter: user=phone
    Call-ID: 54C67BF25EE9-ql502ze7ws29
…
    Contact: <sip:paul@[fd1b:81ba:ac00:25e7:204:13ff:fe76:4df8]:5060>;reg-id=1
        Contact URI: sip:paul@[fd1b:81ba:ac00:25e7:204:13ff:fe76:4df8]:5060
            Contact URI User Part: paul
            Contact URI Host Part: [fd1b:81ba:ac00:25e7:204:13ff:fe76:4df8]
            Contact URI Host Port: 5060
…
Message Body
    Session Description Protocol
        Session Description Protocol Version (v): 0
        Owner/Creator, Session Id (o): root 1747227651 1747227651 IN IP6
          fd1b:81ba:ac00:25e7:204:13ff:fe76:4df8
            Owner Username: root
            Session ID: 1747227651
            Session Version: 1747227651
            Owner Network Type: IN
            Owner Address Type: IP6
            Owner Address: fd1b:81ba:ac00:25e7:204:13ff:fe76:4df8
        Session Name (s): call
        Connection Information (c): IN IP6 fd1b:81ba:ac00:25e7:204:13ff:fe76:4df8
            Connection Network Type: IN
            Connection Address Type: IP6
            Connection Address: fd1b:81ba:ac00:25e7:204:13ff:fe76:4df8
```

The UDP packets conveying the SIP protocol would also be conveyed as IPv6 traffic.

2.3 IPv6 and RTP

In the world of VoIP, audio signals are conveyed using a stream of RTP network packets each containing a number of audio samples. These RTP packets are encapsulated in UDP packets which in turn are encapsulated in IP packets and then ethernet frames.

When interacting with a device expecting IPv6 RTP, the RTP and UDP packets remain unchanged (except for small difference in computing a UDP checksum) but the underlying IP packet changes from an IPv4 to an IPv6 packet and the Ether Type field in the ethernet frame also changes. The IPv6 packet naturally enough now includes 16 octet IPv6 addresses rather than 4 octet IPv4 addresses.

2.4 Dual Stack

The term dual stack is sometimes used to indicate systems or applications that are capable of processing a mixture of both IPv6 calls and IPv4 calls. Once a particular call is set up, incoming RTP for the call would normally be either received by the end system exclusively as IPv6 or exclusively as IPv4 as agreed in the SIP negotiation (however SIP procedures such as reinvite or refer specifying differing requirements could cause this to change).

2.5 IPSec

IPSec is an element of the IPv6 stack that allows encryption of IP packets at a low level (underlying IP packets). Normally with VoIP systems when encryption and/or authentication is required, this occurs at a higher level using secure RTP and/or SIP TLS facilities. IPSec is not currently supported for ProsodyX. If set up for the underlying operating system it could in principle be used with ProsodyS based systems.

3 Code Changes required for IPv6

Note for Windows a relatively recent development system is required to gain access to IPv6 related headers and libraries (such as support for standard C library call inet_ntop).

3.1 Important Note Regarding ProsodyX Media Addresses

One important difference to note for ProsodyX applications which handle IPv6 is that:

- from IPv4 perspective the base ProsodyX card and all its DSPs shared a single IPv4 address which is also the media endpoint address for RTP
- from IPv6 perspective each DSP has its own media endpoint IPv6 address (which may be discovered through the resource manager API call acu_prosody_ip_get_endpoint_address).

Thus to obtain media IPv6 address for DSP #0 one might invoke:

```
ACU_PROSODY_IP_ENDPOINT_ADDRESS_PARMS endpoint;

INIT_ACU_STRUCT(&endpoint);

endpoint.card_id = cardId;
endpoint.device  = ACU_CARD_ENDPOINT_DSP;
endpoint.index   = 0;

acu_prosody_ip_get_endpoint_address(&endpoint);
```

3.2 SIP API

With an IPv6 application, the media address supplied in:

```
ACU_MEDIA_OFFER_ANSWER
```

needs to set:

```
connection_address.address_type = ACU_IPv6
```

As noted above, for ProsodyX the specified connection_address must be the media address of the DSP module (not the IPv6 address of the base card which is different).

The textual format of an IP address as passed to SIP API (SIP URI) can hold:

- IPv4
- IPv6
- hostnames.

Hostnames and IPv4 are already familiar through IPv4 applications:

- <sip:fred@192.168.200.32>
- <sip:fred@somename.com>

An IPv6 address would look similar to this:

- <sip:fred@[fd94:232d::1234]>

The SIP specifications require that IPv6 addresses are surrounded by square brackets. A valid URI must be used instead of merely just a plain IPv6 address.

3.3 Call Control (using MHP) API

For new applications it is generally recommended to use the extended SIP API rather than the legacy method of making SIP calls through call control API with assistance of Media Handler Plugin (MHP).

However if MHP facilities have to be used, then they can be used to make and receive IPv6 as well as IPv4 SIP calls. MHP calls can be invoked with application allocated TiNG media resources (iptel_port mode where application has previously invoked call_open_ iptel_port) or (for ProsodyX) with media resource allocation delegated to MHP and bearer channel linked to TDM timeslot (generic timeslot mode).

3.3.1 Using iptel_port mode

In this mode of operation, the application uses the TiNG API to create VMPTx and VMPRx media resources.

When making an outgoing call the choice of whether the SDP in the INVITE request will specify IPv6 is determined by whether the type of the VMPRx is set to kSMVMPrxTypeIPv6. This is a decision taken by the application.

When receiving an incoming call the application doesn't know in advance if the call will be for IPv4 or IPv6. When invoking call_openin the vmp identifiers should be set to special value `ACU_WILL_PROVIDE_VMP`:

```
inParms.unique_xparms.sig_iptel.vmprxid = ACU_WILL_PROVIDE_VMP;
inParms.unique_xparms.sig_iptel.vmptxid = ACU_WILL_PROVIDE_VMP;
```

It must then wait until it is able to query the ipv6_media field in uniquex_iptel. This tells the application whether the offer in the received INVITE message requested IPv6. If this field is non-zero it must create an IPv6 VMPRx for use in call_incoming_ringing(), call_progress() and call_accept().

In general use, when a VMPRx is created it is optional to specify an explicit ipv6_address with which VMPRx is associated. However when an VMPRx is created to be passed to MHP for use in iptel_port mode MHP application - it is mandatory to specify an IPv6 address when creating the VMPRx (normally obtained from call to acu_prosody_ip_get_endpoint_address for ProsodyX).

In OUT_XPARMS structure, the bracketed form of IPv6 address must be used for "originating_addr" and destination_addr, so for example when setting up an MHP originating address one could do:

```
sprintf(&out_xparms.originating_addr[0],"<sip:[%s]>",&ourSIPAddr[0]);
```

3.3.2 Using generic timeslot mode

To require an outgoing call to use IPv6 the ipv6_media field in uniquex_iptel should be set to a non-zero value. This will instruct the MHP to create an IPv6 VMPRx and vmptx.

Incoming calls are handled automatically based on the information received in an INVITE request.

3.4 TiNG API

3.4.1 IPv6 Endpoint Creation and Status Monitoring

ProsodyS/X can work with both IPv4 and IPv6 RTP streams, but at any time an individual RTP endpoint is either in IPv4 or IPv6 mode.

When creating a VMPRx, it is necessary to specify whether it is to accept IPv6 RTP or IPv4 RTP using the type field.

If an explicit listen address is specified, it may be either a global scope IPv6 address or a link-local IPv6 address.

In contrast to a IPv4 type VMPRx, when a VMPRx status event is processed, an application should expect to handle:

kSMVMPrxStatusGotPortsIPv6

kSMVMPrxStatusNewSSRCIPv6

instead of:

kSMVMPrxStatusGotPorts

kSMVMPrxStatusNewSSRC

and should access ports_ipv6 and ssrc_ipv6 elements of SM_VMPRX_STATUS_PARMS union.

A VMPTx does not have inherent IPv6 or IPv4 type until it is configured with destination address, so is created as normal using sm_vmptx_create.

T.38 FMP endpoints have similar IPv6 usage properties.

3.4.2 VMPTx IPv6 Configuration

When the VMPTx is configured, this should be done with new call sm_vmptx_config_ipv6.

The equivalent field to TOS_RTP for IPv6 is bits 20..27 of destination_rtp.sin6_flowinfo which will get placed in the RTP IP packets Traffic Class 8 bit field. This functionality is not currently available under Windows.

NOTE

Note if a source address is specified when configuring an VMPTx for IPv6, both source and destination addresses must be of same scope so either both global or both link local.

If a loopback is required the IPv6 form of loopback address should be used.

3.4.3 TiNG IPv6 Session with ProsodyS/ProsodyX

The network protocol used to convey API instructions (TiNG messages) between TiNG library and ProsodyS/ProsodyX may be either IPv6 or IPv4, the variant used depending on type of address present in locator passed to call that initiates the session (e.g. sm_open_prosody_x). If an IPv6 address is specified, the bracketed form of the address should be used. Note also for ProsodyX, the address used is the IPv6 of the base card (not that of a DSP).

Similar considerations apply to use of TiNG loading tools such as kloadx and modload.

If application session with TiNG is initiated with acu_open_prosody then session will take place over address that would be returned by in ip_address field of ACU_CARD_INFO_PARMS if acu_get_card_info were invoked (IPv4 unless use of IPv4 is disabled on card in which case it will take place over IPv6).

4 Configuring and Deploying Systems for IPv6

4.1 ProsodyX

ProsodyX cards will require flash upgrade against 6.6 distribution in order to be fully IPv6 capable. Flash upgrade is done in the normal way using Aculab configuration tool (Act) or prosody_ip_card_flash command.

IPv6 is only supported by rev3 ProsodyX cards and HA chassis, IPv6 is not supported on Prosody 1U Enterprise chassis.

The Act or prosody_ip_card_mgr command may be used to configure a ProsodyX card to operate in:

- IPv4 mode only
- IPv6 mode only
- Dual Stack IPv4 and IPv6 mode

If IPv6 functionality is not required, it is recommended to configure card in IPv4 only mode so that DSP resource for IPv6 support is not allocated. If IPv6 is enabled there will be less DSP memory available so application channel counts may be reduced for memory bound applications.

The mode selection is achieved when adding ProsodyX card using Act through selecting IPv4 and/orIPv6 in "IP stack options" area (or equivalent command line flags when using prosody_ip_card_mgr command).

If IPv6 is enabled then (with boot card checked) the IPv6 address assignment behaviour can be set to one of:

- Auto configure
- Prefix
- Manual

Normally the "Auto configure" mode is appropriate but if there are special requirements for address assignment one of the other two modes may be employed (in which case a gateway router address may also be explicitly specified). For manual mode individual unique addresses must be assigned to base card and DSPs, all constructed with common prefixes of specified subnet prefix length in bits.

4.2 ProsodyS

The IPv6 address used for ProsodyS (as an RTP media address) will just be one of the IPv6 addresses assigned to the network interfaces for the system (as listed by ipconfig/ifconfig). If multiple global addresses are assigned to an interface, it is up to the application developer to select the most suitable one to use.

If a remote ProsodyS server is configured through specifying the global IPv6 address of the system on which it is hosted, there should just be one NIC on that system that could be used route messages back to a TiNG application and if possible it should just have a single global IPv6 address. If the system is not configured in this way, your application might have trouble connecting to the remote ProsodyS server - see "Troubleshooting Tips" section for more information.

4.3 SIP

If a pre-existing SIP configuration file is to be used for an IPv6 capable system it will need to be modified to add extra IPv6 specific configuration information, namely:

UseIPv6 = 1

UDPListenAddress = [::]

TCPListenAddress = [::]

TLSListenAddress = [::]

In addition, SIP IPv6 operation requires the presence of a file (created when a 6.6 distribution is installed) with the name "sip_enable_ipv6.cfg" the in same cfg sub-directory with a line present as follows:

enable = 1

5 Troubleshooting Tips

Cards not coming up

If remote cards set up with IPv6 autoconfigured addresses do not come up - verify IPv6 router (or radvd) is attached to network and running - are its IPV6 ICMP RA packets visible when sniffing network with wireshark?

IPv6 Packets not reaching DSPs

For ProsodyX, the pxdiagutil command can be used to list VMPTx and VMPRx objects associated with a module to see if they are transmitting or receiving packets. It can also be used to list the IPv6 addresses of the DSP modules. See TiNG documentation:

TiNG/pubdoc/prog_pxdiagutil.html

For diagnostic purposes, the TiNG test tools rtpplay and rtprec can be used to generate or receive an IPv6 RTP stream with similar parameters to those used for iPv4, however the rtprec needs a "-6" flag to indicate a VMPTx of IPv6 type should be created. An example of an IPv6 rtpplay command line is given below:

```
rtpplay -x x:[fdcd:fb9b::a43b]/mykey -d fdcd:fb9b::a6c6 -p 50000 -F a -m 8 -c a s.alaw
```

IPv6 Application fails to connect to remote ProsodyS

The underlying link between a TiNG application and a ProsodyS server is based on the UDP protocol and depends upon the source IP address used in UDP replies being the same as the destination UDP IP address used in requests. When ProsodyS is located in a remote system, some system configurations may cause this not to be the case and the following may be reported in TiNGtrace output:

```
Cannot connect to <IPv6 address>:cardinfod
```

A wireshark capture for UDP port 2030 may be used to ascertain if a "wrong" source address is the problem in such a case. If so, its likely the problem can be rectified by making sure the ProsodyS host system only has a single NIC active on the relevant link and it has a single global IPv6 addresses assigned to it. If multiple global IPv6 addresses are listed, its likely that selecting the first in the list as the address to specify by peer ProsodyS using systems will resolve the issue. Alternatively use of an IPv4 address for application/ProsodyS interaction could be considered.

The “assp” ProsodyS configuration parameter documented in the “Configuration file” section of the ProsyS User Guide can also be used to control source address used in these UDP packets and can be another way to resolve such problems.