Net::SIP

Steffen Ullrich
GeNUA mbH

Was ist Net::SIP

- Implementation von SIP (VoIP) in perl
- auch ein bißchen SDP sofern für SIP erforderlich
- und auch etwas RTP damit man was zum Testen hat
- RFCs 3261, 2327, 3263, 2617,...

Was kann man damit machen

- SIP proxy
- NAT für SIP
- 3pcc (3rd party call control)
- Anrufbeantworter
- ...

Was gibt es für Klassen

- Packet Level
  - Net::SIP::Packet
  - Net::SIP::Request
  - Net::SIP::Response
  - Net::SIP::SDP
- Useragent Level
  - Net::SIP::Dispatcher*
  - Net::SIP::Endpoint*
  - Net::SIP::StatelessProxy
  - Net::SIP::Registrar
  - Net::SIP::NATHelper*
- High Level: Net::SIP::Simple

Beispiel: Listen+Save

```
 1 use strict;
 2 use Net::SIP ':all';
 3 # create user agent
 4 my $uas = Net::SIP::Simple->new(
 5     leg => '127.0.0.1:9999'
 6 ) || die;
 7
 8 # listen for incoming call
 9 my $stop;
10 $uas->listen(
11     # echo back and save received RTP in audio.data
12     init_media => $uas->rtp( 'recv_echo', 'audio.data' ),
13     # set $stop once the first call got completed
14     cb_cleanup => \$stop,
15 );
16 # wait until $stop is true
17 $uas->loop( \$stop );
```

Beispiel: Call+Send

```
 1 use strict;
 2 use Net::SIP;
 3 # create user agent
 4 my $uac = Net::SIP::Simple->new(
 5     from => 'me@example.com',
 6     outgoing_proxy => 'proxy:5060',
 7 ) || die;
 8
 9 # invite other party
10 my $stop;
11 $uac->invite(
12     to => 'you@example.com',
13     # send data.pcmu-8000 two times
14     init_media => $uas->rtp( 'recv_send', 'audio.data',2 ),
15     # set $stop to true if done sending RTP
16     cb_rtp_done => \$stop,
17 );
18 $uac->loop( \$stop ); # wait until $stop gets true
```

Status

- Beta, aber nutzbar
- ausführlich dokumentiert (Tutorial wäre gut)
- einiges an Tests
- einiges an Beispielen und Applikationen, u.a. einfacher Anrufbeantworter
- aktiv in Development

Ende