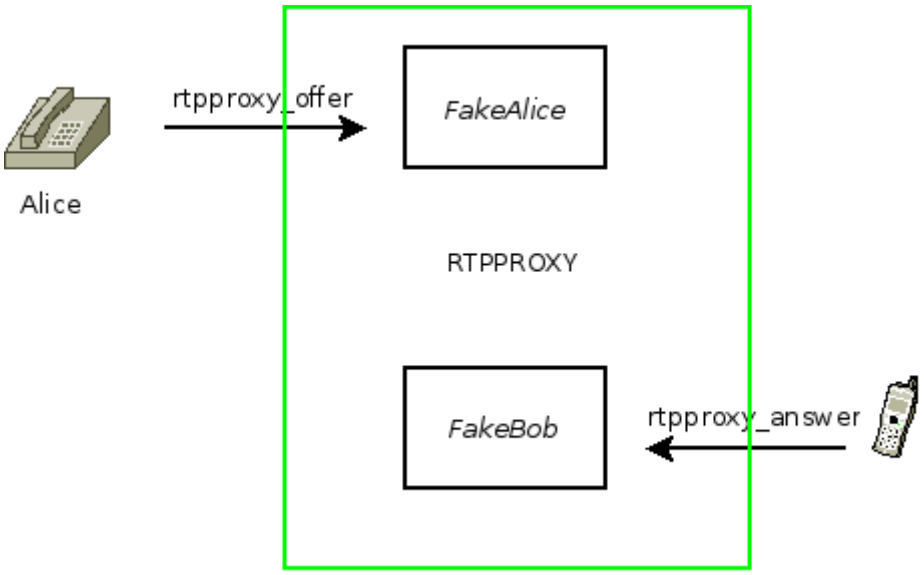
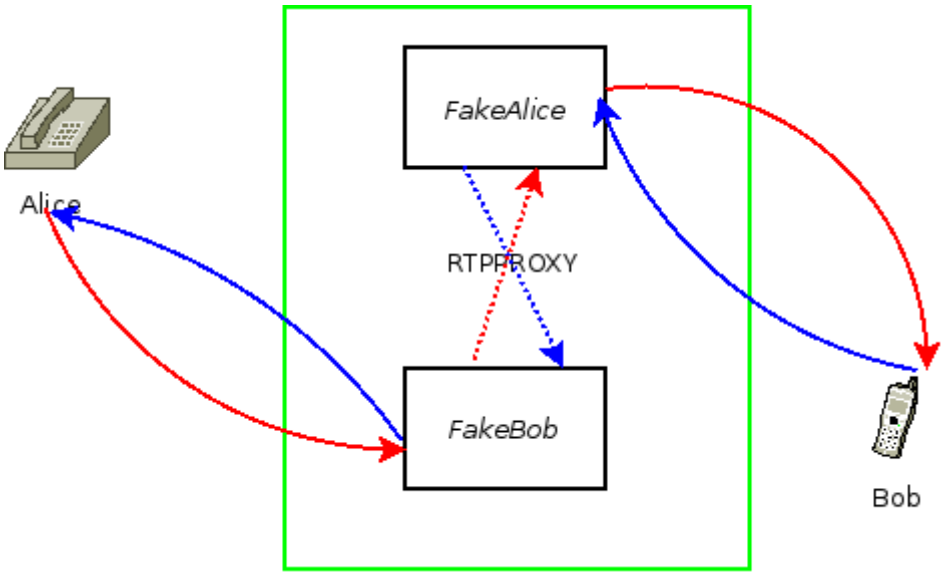
How does this work.

Assume that Alice is a «dumb» client with only PCMA/PCMU available while Bob is a rich client with a full set of codecs (with a G.729 among them which is a codec preferred by him).

Let's assume that Alice calls Bob via RTP proxying device.



Alice sees only FakeBob socket opened within rtpproxy, and Bob sees only FakeAlice socket. Now let's see how media traffic will be routed.



Alice sends everything to FakeBob who as she believes is a real person, Bob. RTPproxy resends this traffic from FakeAlice socket so Bob, who believes this is a real Alice and sends everything back to FakeAlice. So what we're doing is a recreation of a symmetric RTP channel.

Transcoding.

Remember, Alice device is dumb, while Bob's one is feature-rich.

Alice offers the following payload types - PCMA, CN (confort noise), and DTMF (dynamic type #101). Bob offers — PCMA, PCMU, DTMF (dynamic type #101), and G.729. What we really want is to allow Bob to send and receive G.729.

Alice's INVITE contains the following SDP:

```
v=0
o=FreeSWITCH 1353934215 1353934216 IN IP4 192.168.1.18
s=FreeSWITCH
c=IN IP4 192.168.1.18
t=0 0
m=audio 40638 RTP/AVP 8 101 13
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=ptime:20
a=nortpproxy:yes
```

We rewrote this SDP and replace CN (payload #13) with G.729 (payload #18) by altering SDP in the following way:

```
v=0
o=FreeSWITCH 1353934215 1353934216 IN IP4 192.168.1.18
s=FreeSWITCH
c=IN IP4 192.168.1.18
t=0 0
m=audio 40638 RTP/AVP 18 101
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=ptime:20
a=nortpproxy:yes
```

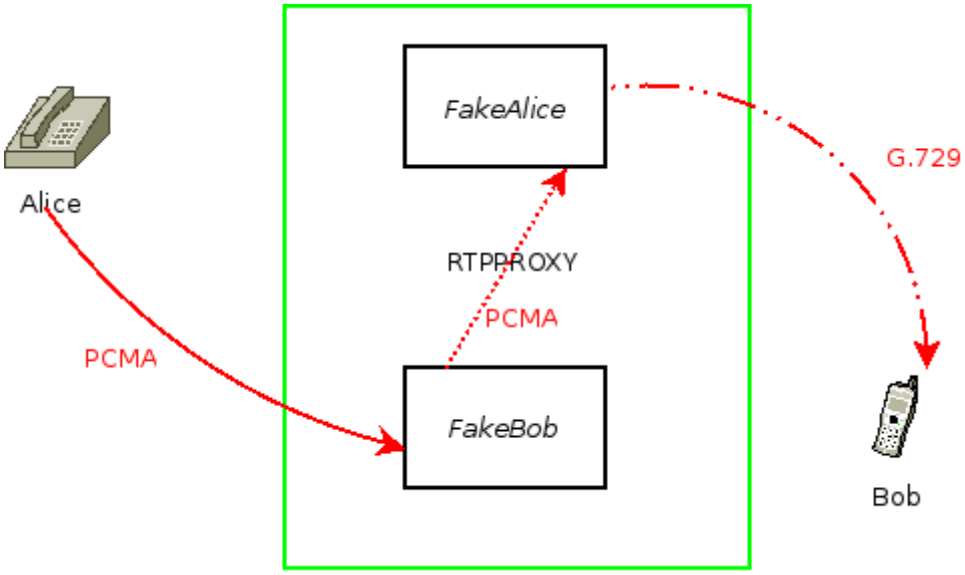
So preferred codec will be G.729 followed by DTMF.

When Bob answers we'll rewrite his SDP again and replace #18 with #8. This should be done on a higher level than rtpproxy (with OpenSIPs text processign options, or with B2BUA) and this is outside the scope of this document.

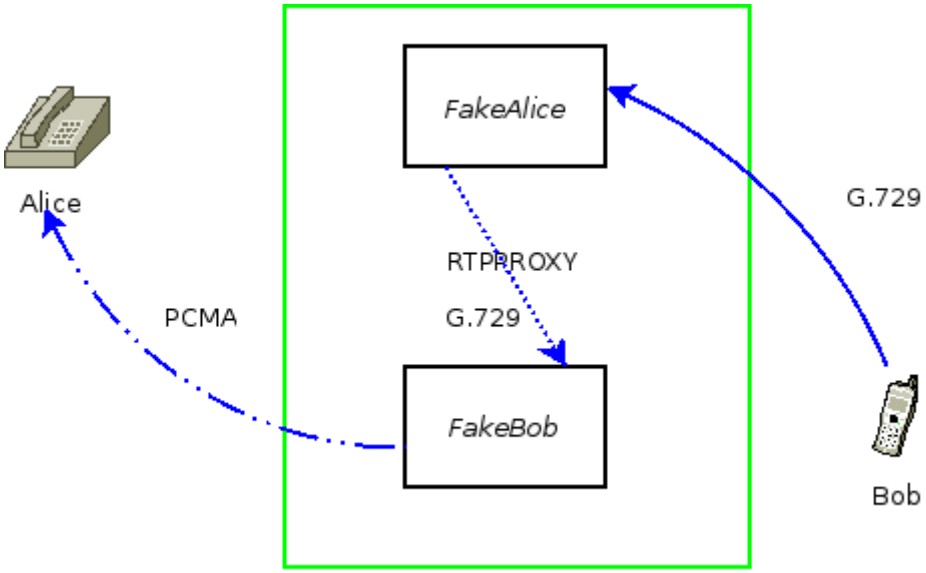
Currently erlrtpproxy transcodes payload before sending further. So an option for transcoding a stream to Bob must be set within rtpproxy_offer command (which creates socket which actually represents Alice). Likewise an option for stream transcoding to Alice must be set within rtpproxy_answer command (which creates socket which actually represents Bob).

Assuming that we did all the SDP alterations properly let's take a look how our parties are going to exchange media streams.

Alice starts sending PCMA which got transcoded to G.729 before sending to Bob:



Bob starts sending G.729 which got transcoded to PCMA before sending to Alice:



tbc...