emulator

Name

`emulator` — utility intended to test how network losses affects speech quality in VoIP-based applications.

Synopsis

emulator `--list-codecs`

emulator [`-i--input-file`]*`input_file.wav`* [`-o--output-file`]*`output_file.wav`* [`-c--codec`]*`codec_name`* [`-b--bitrate`]*`codec_bitrate`* [`-l--loss`]*`lost_pct`* `--p00`*`lost_pct`* `--p10`*`lost_pct`* `--burst-ratio`*`value`* [`-f--fpp`]*`frames_per_packet`* [`-p--plc`]*`algo`* [`-q--speex-quality`]*`value`* [`-Q--speex-vbr-quality`]*`value`* `--log-level`*`level`* `--bucket-size`*`size`* `--bucket-size`*`size`* [`--bw--bandwidth`]*`value`* `--show-stats`

emulator `--help`

DESCRIPTION

emulator is a simple utility intended to test how network losses affects speech quality in VoIP-based applications. Experimenter can set up encoder, channel and decoder options, such as codec type, loss rate, bandwith and one of the packet loss suppression algorithm on the receiver side.

emulator consists of three parts. Encoder reads data from input file, encodes it with selected codec, packs data into RTP-packets. Channel obtains packets from encoder, emulates losses and delays and send packets to the next party. Decoder receives data from channel, performs packet loss concealment and decoding, then stores data into the output file.

A few words about channel emulator algorithms. Channel consists of two blocks: *random loss emulator* and *bandwidth limitation emulator*. The random loss emulator can emulate random uncorrelated losses with `-l` option or losses modelled with markov chain.

The markov chain can be defined by two alternative means. The first one is to to define two options: `--p10` which specifies packet loss probability (in percent) in the case when previous packet has not be lost, and `--p00` which specifies the same in the case when previous packet has been lost as well. The second variant consist in usage `-l` option along with `--burst-ratio`. Burst ratio is defined as ratio of the average length of observed bursts in an arrival sequence to the average length of bursts expected for the network under random loss with no correlation between packets.

Bandwith limitation is emulated with leaky bucket algorithm. Implementation receives two options: `--bucket-size` in packets and `--bandwidth` in bits per second (bps) or packets per second (pps).

OPTIONS

Encoder options

`-i`, `--input-file` *`file.wav`*

Specify input (referenced) file. File must be in .WAV format, and its sampling rate must be equal with codec sampling rate (8000 samples per second in most cases).

`-c`, `--codec` *`codec`*

Specify codec name. Allowed codec names must be displayed with `--list-codecs` option.

`-f`, `--fpp` *`1..10`*

Specify number of encoder frames per one RTP packet.

`-q`, `--speex-quality` *`0..10`*

Specify speex encoder quality (obviously this option makes sence only with "speex" codec).

`-Q`, `--speex-vbr-quality` *`0..10`*

Specify speex encoder quality in VBR quality (floating point number). Setting up this option causes the encoder to switch the Speex encoder in VBR mode with given quality. This option is applicable only if PJMEDIA has Speex VBR support.

`-b`, `--bitrate` *`codec_bitrate`*

AMR, AMR-WB and G723 codecs define encoding mode using ‘bitrate’ value. Below values are acceptable:

AMR: 4750, 5150, 5900, 6700, 7400, 7950, 10200, 12200
AMR-WB: 6600, 8850, 12650, 14250, 15850, 18250, 19850, 23050, 23850
G723: 5300, 6300

If PJSIP has speex VBR/ABR support then `bitrate` option can accept any integer as value.

Channel options

`-l`, `--loss` *`loss_pct`*

Specify channel loss rate (floating number from 0 to 100). Default value is 0.

`--p00` *`loss_pct`*

Specify p00 parameter of the markov chain loss model (floating number from 0 to 100). Default value is 0.

`--p10` *loss_pct*

Specify p10 parameter of the markov chain loss model (floating number from 0 to 100). Default value is 0.

`--burst-ratio` *ratio*

Specify burst ratio of the markov chain loss model. Used along with `--loss` option.

`--bucket-size` *N*

Specify bucket size in the leaky bucket traffic shaping model.

`--bw`, `--bandwidth` *XXbps|YYpps*

Specify bandwidth in the leaky bucket traffic shaping model. Note that full size of the one RTP packet consists of these addends: IP header size (20 bytes), UDP header size (8 bytes), RTP header size (12 bytes) and payload size itself.

Decoder options

`-p`, `--plc` empty|repeat|smart|noise

Specify packet loss concealment algorithm. Allowed values are:

empty: lost frames replaced with empty ones;
repeat: lost frames replaced with last received frame;
smart: PLC based on WSOLA algorithm or built-in speex PLC methods for Speex;
noice: lost frames replaced with white noise.

Note that "smart" (WSOLA-based) PLC implementation adds some zeros in the beginning of the frame (about half of samples per one packet) which may give some unwanted side effects as unreasonable PESQ decreasing is voice quality tests.

`-o`, `--output-file` *file.wav*

Specify output (degraded) file.

Miscellaneous options

`--show-stats`

Display short emulation statistics: total packets sent, packets lost, packets received and the rate of lost packets during current emulation.

`--log-level` 0..6

Define log level verbosity. Zero means "display nothing", six means "push detailed log on the stdout".

`--help`

Display this man page.

EXAMPLES

Get the output signal in the "real channel" with short stats at the end of emulation

```
$ emulator --input-file i.wav --codec speex/8000 --speex-quality 8 --fpp 8 \
>   --loss 5 --bw 4.9kbps --bucket-size 10 \
>   --plc smart --output-file o.wav --show-stats
Emulation statistics
  total packets sent: 63
        packets lost: 35
    packets received: 28
        loss percent: 55.56
```

FILES

nothing special is used

AUTHOR

Roman Imankulov <`roman@netangels.ru`>

SEE ALSO

sox(1), speexenc(1), speexdec(1),