

Debugging_Event_Socket_Message

You may want to see what's going back and forth between FreeSWITCH and your application when you are developing an ESL application. Sure there are many ways to do this, here is just one:

The idea is to put a Man in the Middle, and he tells us whatever he saw. Thanks for the UNIX art to make this so simple:

You need the help of two temporary files, be sure to select two unique names:

```
echo > /tmp/a
echo > /tmp/b
tail -f /tmp/b &
tail -f /tmp/a | nc localhost 8021 | tee /tmp/b | nc -l -k 8022 | tee /tmp/a
```

If FreeSWITCH listens on 8021, open a terminal and type the above commands, then make your ESL application connect to localhost 8022 you should be able to see all the messages going back and forth.

The Middle Man is actually a B2B socket (just like FreeSWITCH is a B2BUA). The A-leg connect to FreeSWITCH and the B-leg create a socket waiting your ESL app to connect. To understand the last line you need to know the basics of UNIX (STDIN/STDOUT/STDERR) and nc (netcat).

You will need manually to kill the "tail -f /tmp/b" process after debug. To avoid this, you might like the following way:

Think about the UNIX "tee" is to copy STDIN to STDOUT and other files if any. We wouldn't need to 3rd line if it can also copy the STDIN to STDERR. OK, here is a simple tee2:

```
/* tee2 - read from standard input and write to standard output,
standard error and a file.
Useful when you redirect the stdout but still want to keep an eye on
the output.

Want to write to more than one file? Just don't forget the UNIX art
# echo blah | tee2 /tmp/1 | tee /tmp/{2,3,4,5,6}

/* Author: Seven Du */

#include <stdio.h>
#include <stdlib.h>

int main (int argc, char **argv)
{
    char c;
    char *file_name;
    FILE *file;
    if (argc < 2) {
        fprintf(stderr, "Usage: tee2 <file_name>\n");
        exit(1);
    }

    file_name = argv[1];

    if(! (file = fopen(file_name, "w")) ){
        fprintf(stderr, "Error write file %s !\n", file_name);
        exit(2);
    }
}
```

Debugging_Event_Socket_Message

```
while((c = getchar()) != EOF) {
    putc(c, file);
    putc(c, stdout);
    putc(c, stderr);

    if(c == '\n') {
        fflush(file);
        fflush(stdout);
        fflush(stderr);
    }

}

fclose(file);
exit(0);
}
```

And just compile it:

```
gcc -o tee2 tee2.c
sudo cp tee2 /usr/bin
```

And then do this:

```
echo > /tmp/a
echo > /tmp/b
tail -f /tmp/a | nc localhost 8021 | tee2 /tmp/b | nc -l -k 8022 | tee /tmp/a
```

Also it maybe also easy to make a perl or ruby version of tee2 :). Any better way to do this? Feel free to edit this page.