Bip - Bug #637

Segfault linked to NICK handling

2018-10-31 14:25 - raph raph

Status:	New	**Start date:**	2018-10-31
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
Patch Available:		**Branch:**	
Found in Versions:		**Security:**	
Confirmed:	No	**Help Needed:**	

Description

Hello,
I've had several crashes using bip 0.9.0-rc1-git

For example:

```
(gdb) bt

#0  0x00005555555638dc in write_line (cn=cn@entry=0x0, line=line@entry=0x5555559427b0 "NICK Trou\r
\n") at src/connection.c:430

#1  0x000055555556b6bc in irc_line_write (l=<optimized out>, c=0x0) at src/line.c:41

#2  0x0000555555566d1e in irc_001 (server=0x55555593e970, server=0x55555593e970, line=0x5555559357
b0) at src/irc.c:173

#3  irc_dispatch_server (bip=<optimized out>, server=0x55555593e970, line=0x5555559357b0) at src/i
rc.c:453

#4  0x000055555556900d in irc_dispatch (bip=bip@entry=0x7fffffffb4b0, l=l@entry=0x55555593e970, li
ne=line@entry=0x5555559357b0) at src/irc.c:1260

#5  0x000055555556ac85 in bip_on_event (bip=bip@entry=0x7fffffffb4b0, conn=0x5555558b8750) at src/
irc.c:2490

#6  0x000055555556af73 in irc_main (bip=0x7fffffffb4b0) at src/irc.c:2565

#7  0x000055555555b4a0 in main (argc=<optimized out>, argv=<optimized out>) at src/bip.c:1359
```

I've also another coredump (but for some reason, gdb does not get lines correctly):

```
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x000055e45d24d8dc in ?? ()
(gdb) info reg
rax            0x55e45d759a80   94439308892800
rbx            0x0      0
rcx            0x7f45d71b5b00   139937938365184
rdx            0x12     18
rsi            0x55e45d7599e0   94439308892640
```

```
rdi            0x55e45d759a80   94439308892800
rbp            0x0      0x0
rsp            0x7ffc0674a1c0   0x7ffc0674a1c0
r8             0xeec0   61120
r9             0x20     32
r10            0x747e21756f725420       8394183543729837088
r11            0x312e383740756f72       3543831766742953842
r12            0x7ffc0674a220   140720416793120
r13            0x1      1
r14            0x0      0
r15            0x8      8
rip            0x55e45d24d8dc   0x55e45d24d8dc
eflags         0x10206  [ PF IF RF ]

(gdb) x/1s $rdi
0x55e45d759a80: ":Trou NICK Trou\r\n"

(gdb) x/1s $rsi
0x55e45d7599e0: ":Trou NICK Trou\r\n"
```

But i'm not quite sure why it crashes.

The following patch should help if there's a race (between increment and realloc) but I think it's unlikely to be the problem.

```
diff --git a/src/irc.c b/src/irc.c
index f46f4dd..08c9cb9 100644
--- a/src/irc.c
+++ b/src/irc.c
@@ -575,10 +575,11 @@ static void bind_to_link(struct link *l, struct link_client *ic)
        int i = l->l_clientc;

        LINK(ic) = l;
-       l->l_clientc++;
-       l->l_clientv = bip_realloc(l->l_clientv, l->l_clientc *
+       l->l_clientv = bip_realloc(l->l_clientv, (l->l_clientc + 1) *
                        sizeof(struct link_client *));
        l->l_clientv[i] = ic;
+    /* Increment after, to avoid any race condition */
+       l->l_clientc++;
 }

 void unbind_from_link(struct link_client *ic)
```

History

#1 - 2018-10-31 15:05 - raph raph

I think it may be related to a logic problem due to SSL certs trust:

```
31-10-2018 14:49:49 SSL cert check: now at depth=0
31-10-2018 14:49:49 Subject: /CN=XXXX
31-10-2018 14:49:49 Issuer: /C=US/O=Let's Encrypt/CN=Let's Encrypt Authority X3
31-10-2018 14:49:49 Basic mode; peer certificate NOT in store, rejecting it!
31-10-2018 14:49:49 ERROR: SSL cert check failed at depth=0: certificate rejected (28)
140737354060736:error:1416F086:SSL routines:tls_process_server_certificate:certificate verify failed:../ssl/st
atem/statem_clnt.c:1230:
31-10-2018 14:49:49 ERROR: Certificate check failed: certificate rejected (28)!
31-10-2018 14:49:49 ERROR: Error on fd 1 (state 9)
31-10-2018 14:49:49 DEBUG: New socket still not connected (2)
31-10-2018 14:49:49 ERROR: [XXX] read_lines error, closing...
```

then i get later a trust request for the certificate BUT at the same time, bip connects to another server, with a valid certificate

```
14:51 [XXX] !b.i.p This server SSL certificate was not accepted because it is not in your store of trusted cer
tificates:
14:51 [XXX] !b.i.p Subject: /CN=XXX
14:51 [XXX] !b.i.p Issuer:  /C=US/O=Let's Encrypt/CN=Let's Encrypt Authority X3
[...]
14:51 [XXX] !b.i.p WARNING: if you've already trusted a certificate for this server before, that probably mean
s it has changed.
14:51 [XXX] !b.i.p If so, YOU MAY BE SUBJECT OF A MAN-IN-THE-MIDDLE ATTACK! PLEASE DON'T TRUST THIS CERTIFICAT
E IF YOU'RE NOT SURE THIS IS NOT THE CASE.
14:51 [XXX] !b.i.p Type /QUOTE BIP TRUST OK to trust this certificate, /QUOTE BIP TRUST NO to discard it.
```

then

```
(gdb) bt
#0  0x000055555556470c in write_line (cn=cn@entry=0x0, line=line@entry=0x555555770440 "NICK Trou\r\n") at src/
connection.c:430
#1  0x0000555555556c60c in irc_line_write (l=<optimized out>, c=0x0) at src/line.c:41
#2  0x0000555555567bc4 in irc_001 (server=0x55555575cac0, server=0x55555575cac0, line=0x55555574d090) at src/i
rc.c:173
#3  irc_dispatch_server (bip=<optimized out>, server=0x55555575cac0, line=0x55555574d090) at src/irc.c:453
#4  0x0000555555569ed5 in irc_dispatch (bip=bip@entry=0x7fffffffb480, l=l@entry=0x55555575cac0, line=line@entr
y=0x55555574d090) at src/irc.c:1261
#5  0x000055555556bb91 in bip_on_event (bip=bip@entry=0x7fffffffb480, conn=0x555555743de0) at src/irc.c:2491
#6  0x000055555556bea3 in irc_main (bip=0x7fffffffb480) at src/irc.c:2566
#7  0x000055555555bea8 in main (argc=<optimized out>, argv=<optimized out>) at src/bip.c:1359
(gdb)
```

if look at the connecionts:

```
(gdb) print server->_link_c->link->l_clientv->_link_c->type
$15 = 3
(gdb) print server->_link_c->link->l_clientv[0]->_link_c->type
$16 = 3
(gdb) print server->_link_c->link->l_clientv[1]->_link_c->type

$17 = 3
(gdb) print server->_link_c->link->l_clientv[2]->_link_c->type

$18 = 3
```

#2 - 2018-12-18 04:32 - Arnaud Cornet

I think this happens if you have a server waiting for "trust" and you disconnect a client then reconnect and successfully trigger a server connection. The closing of a client that was only expecting a trust messages is not cleaned up properly.