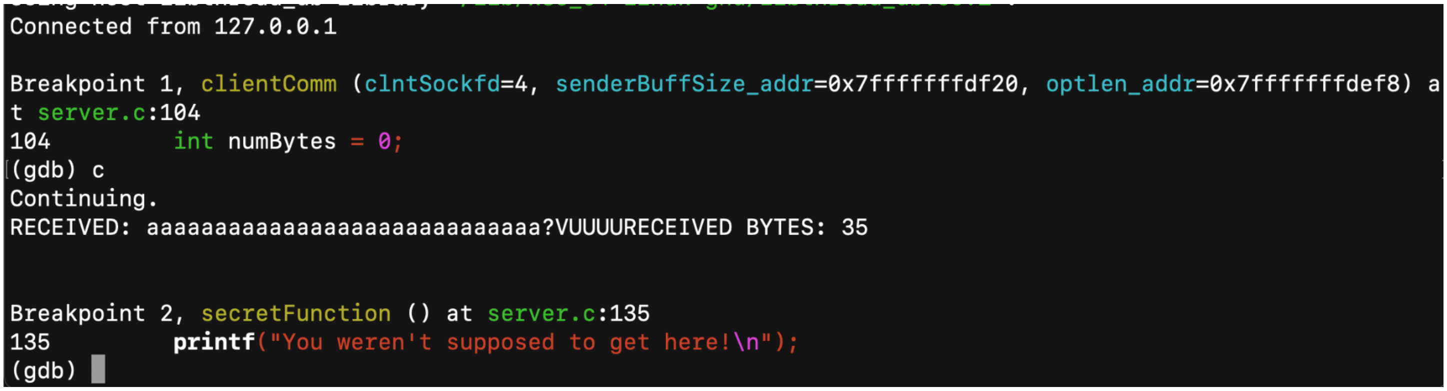
Anish Laddha
Professor Avinash Kak
ECE404
February 19 2024

HOMEWORK 10

Buffer Overflow String: aaaaaaaaaaaaaaaaaaaaaaaaaaaaa\xda\x56\x55\x55\x55\x55\x00\x00

I found this string by first using gdb to find the address of the beginning of secretFunction, which was 0x00005555555556da. Then I reversed its endianness, and added \x to signify that it is a hex. To determine the number of a's i.e. the offset (as each is one byte in size), we can look at the return stack pointer after passing in a value from the client. The number of bytes in between our input and the return stack pointer is the number of ascii characters we include as a buffer, as anything after that will be copied into the value of the return stack pointer.

Then I combined the 2, which allowed me to get the following result:



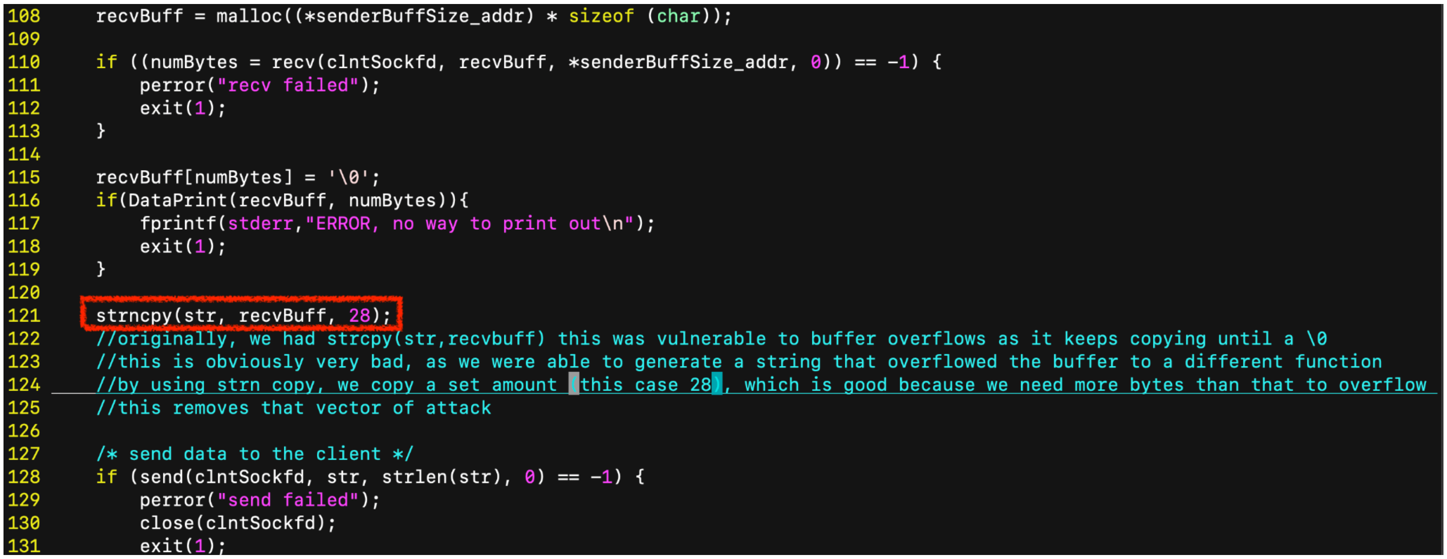
```
Connected from 127.0.0.1

Breakpoint 1, clientComm (clntSockfd=4, senderBuffSize_addr=0x7fffffffdf20, optlen_addr=0x7fffffffdef8) a
t server.c:104
104         int numBytes = 0;
(gdb) c
Continuing.
RECEIVED: aaaaaaaaaaaaaaaaaaaaaaaaaaaaa?VUUUURECEIVED BYTES: 35


Breakpoint 2, secretFunction () at server.c:135
135         printf("You weren't supposed to get here!\n");
(gdb)
```

This image clearly shows my string for buffer overflow tripping the breakpoint I set at secretFunction, showing that the string caused it to go there despite no calls to that function directly.

SERVER.C FIXES



```
    recvBuff = malloc((*senderBuffSize_addr) * sizeof (char));

    if ((numBytes = recv(clntSockfd, recvBuff, *senderBuffSize_addr, 0)) == -1) {
        perror("recv failed");
        exit(1);
    }

    recvBuff[numBytes] = '\0';
    if(DataPrint(recvBuff, numBytes)){
        fprintf(stderr,"ERROR, no way to print out\n");
        exit(1);
    }

    strncpy(str, recvBuff, 28);
    //originally, we had strcpy(str,recvbuff) this was vulnerable to buffer overflows as it keeps copying until a \0
    //this is obviously very bad, as we were able to generate a string that overflowed the buffer to a different function
    //by using strn copy, we copy a set amount (this case 28), which is good because we need more bytes than that to overflow
    //this removes that vector of attack

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

We fix this server.c vulnerability by changing the strcpy to strncpy, which means that only a preset amount (28) of characters would be copied over so a string like mine would be truncated and not be able to induce a buffer overflow.

Log File:

```
New message log:
1
From anishladdha03@gmail.com  Tue Apr  2 18:11:07 2024
 Subject: hello
  Folder: spamFolder                                                          3307

New message log:
2
procmail: Couldn't determine implicit lockfile from "/usr/sbin/sendmail"
From laddhaa@purdue.edu  Tue Apr  2 18:13:22 2024
 Subject: second test
  Folder: /usr/sbin/sendmail -oi laddhaa@purdue.edu                   6120

New message log:
3
From foxnews_37C94726EEA21A17435CEA421612EE444C7F77AF4A1D50C2@response.wc07.net  Tue Apr  2 18:17:35 2024
 Subject: Welcome to Fox News! Thank you for subscribing.
  Folder: spamFolder                                                         24981

New message log:
4
From nyt@service.newyorktimes.com  Tue Apr  2 18:18:37 2024
 Subject: Set the password to your Times account
  Folder: spamFolder                                                         29913

New message log:
5
From nyt@service.newyorktimes.com  Tue Apr  2 18:18:42 2024
 Subject: Set the password to your Times account
  Folder: spamFolder                                                         29899

New message log:
6
From nyt@service.newyorktimes.com  Tue Apr  2 18:18:43 2024
 Subject: Set the password to your Times account
  Folder: spamFolder                                                         30221

New message log:
7
From nyt@service.newyorktimes.com  Tue Apr  2 18:18:44 2024
 Subject: Set the password to your Times account
  Folder: spamFolder                                                         30193

New message log:
8
From nyt@service.newyorktimes.com  Tue Apr  2 18:18:45 2024
 Subject: Set the password to your Times account
  Folder: spamFolder                                                         30234

New message log:
9
From msprvs1=19822z7vU4sHX=bounces-280172-1669@sparkpostmail.com  Tue Apr  2 18:22:26 2024
 Subject: Welcome to Insider Today
  Folder: spamFolder                                                         31754
```