PWNABLE.KR - fd

Lets connect to the server

```
ra@moni~/P/pwnable.kr> ssh fd@128.61.240.205 -p 2222
fd@128.61.240.205's password:
 ____  __    __  ____    ____  ____   _        ___      __  _  ____  
|    \|  |__|  ||    \  /    ||    \ | |      /  _]    |  |/ ]|    \ 
|  o  )  |  |  ||  _  ||  o  ||  o  )| |     /  [_     |  ' / |  D  )
|   _/|  |  |  ||  |  ||     ||     || |___ |    _]    |    \ |    / 
|  |  |  `  '  ||  |  ||  _  ||  O  ||     ||   [_  __ |     \|    \ 
|  |   \      / |  |  ||  |  ||     ||     ||     ||  ||  .  ||  .  \
|__|    \_/\_/  |__|__||__|__||_____||_____||_____||__||__|\_||__|\_|

- Site admin : daehee87@gatech.edu
- IRC : irc.netgarage.org:6667 / #pwnable.kr
- Simply type "irssi" command to join IRC now
- files under /tmp can be erased anytime. make your directory under /tmp
- to use peda, issue `source /usr/share/peda/peda.py` in gdb terminal
You have mail.
Last login: Wed Jun  2 04:57:50 2021 from 106.114.115.94
fd@pwnable:~$
```

After listing the files, we can see there are some privilege restrictions

```
fd@pwnable:~$ ls -la
total 40
drwxr-x---   5 root   fd   4096 Oct 26  2016 .
drwxr-xr-x 115 root   root 4096 Dec 22 08:10 ..
d---------   2 root   root 4096 Jun 12  2014 .bash_history
-r-sr-x---   1 fd_pwn fd   7322 Jun 11  2014 fd
-rw-r--r--   1 root   root  418 Jun 11  2014 fd.c
-r--r-----   1 fd_pwn root   50 Jun 11  2014 flag
-rw-------   1 root   root  128 Oct 26  2016 .gdb_history
dr-xr-xr-x   2 root   root 4096 Dec 19  2016 .irssi
drwxr-xr-x   2 root   root 4096 Oct 23  2016 .pwntools-cache
```

It seems like we cannot read the flag directly

```
fd@pwnable:~$ whoami
fd
fd@pwnable:~$ cat flag
cat: flag: Permission denied
```

And analysing our source code of the binary we get, The RAW CODE is given below,

```
fd@pwnable:~$ cat fd.c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
char buf[32];
int main(int argc, char* argv[], char* envp[]){
    if(argc<2){
        printf("pass argv[1] a number\n");
        return 0;
    }
    int fd = atoi( argv[1] ) - 0x1234;
    int len = 0;
    len = read(fd, buf, 32);
    if(!strcmp("LETMEWIN\n", buf)){
        printf("good job :)\n");
        system("/bin/cat flag");
        exit(0);
    }
    printf("learn about Linux file IO\n");
    return 0;

}
```

From here we can see,

- This binary uses a global variable `buf` of 32 bytes
- This binary gets two inputs from arguments (ie. Filename Arg1)
- It uses `atoi()` to convert string to integer
- It uses `read()` to get the input data
- It uses `strcmp()` and compares it with `LETMEWIN` to display the flag

First inorder to compare the data, we need to store it in the `buf`

Here we are using `read()` to store it in `buf`

To use `read()` properly we need to pass the "file descriptor" correctly

`read()` needs `0` as file descriptor

For more about File Descriptor

To make our file descriptor `fd` as `0`

we should make use of `int fd = atoi( argv[1] ) - 0x1234;`

So if we pass a value of `0x1234` in `argv[1]` we could activate `read()`

Now we need to find value of `0x1234` in `int`

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print(0x1234)
4660
```

If we pass this value, our `read()` function should be opened to pass our data

Now lets pass `LETMEWIN` in our `buf`

```
fd@pwnable:~$ ./fd
pass argv[1] a number
fd@pwnable:~$ ./fd 4660
LETMEWIN
good job :)
mommy! I think I know what a file descriptor is!!
```

Thats it, we got our conditions true

And we get the flag

`FLAG: mommy! I think I know what a file descriptor is!!`