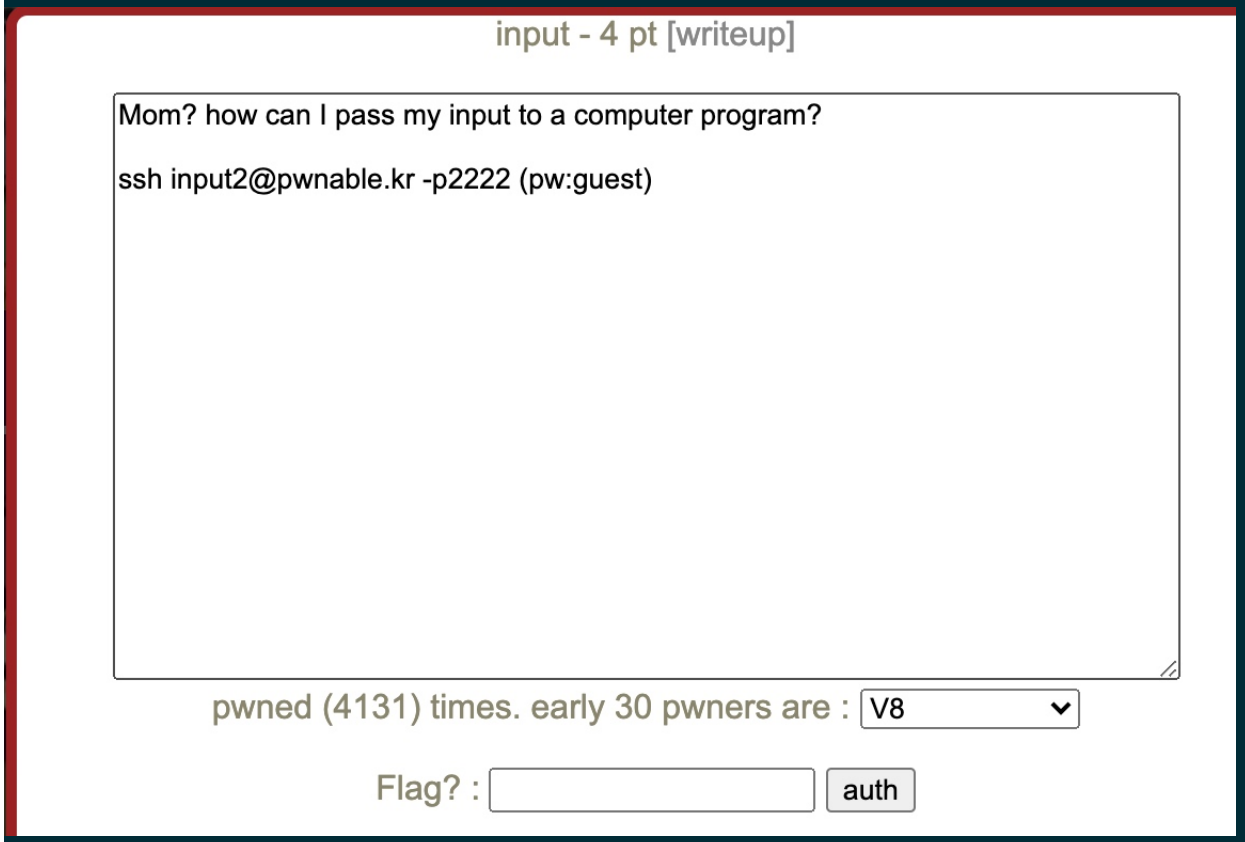


使用ssh input2@pwnable.kr -p2222 进行连接
cat input.c 查看源代码

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>

int main(int argc, char* argv[], char* envp[]){
```

```
    printf("Welcome to pwnable.kr\n");
    printf("Let's see if you know how to give input to program\n");
    printf("Just give me correct inputs then you will get the flag :)\n");

    // argv
    if(argc != 100) return 0;
    if(strcmp(argv['A'],"\x00")) return 0;
    if(strcmp(argv['B'],"\x20\x0a\x0d")) return 0;
    printf("Stage 1 clear!\n");

    // stdio
    char buf[4];
    read(0, buf, 4);
    if(memcmp(buf, "\x00\x0a\x00\xff", 4)) return 0;
    read(2, buf, 4);
        if(memcmp(buf, "\x00\x0a\x02\xff", 4)) return 0;
    printf("Stage 2 clear!\n");

    // env
    if(strcmp("\xca\xfe\xba\xbe", getenv("\xde\xad\xbe\xef"))) return 0;
    printf("Stage 3 clear!\n");

    // file
    FILE* fp = fopen("\x0a", "r");
    if(!fp) return 0;
    if( fread(buf, 4, 1, fp)!=1 ) return 0;
    if( memcmp(buf, "\x00\x00\x00\x00", 4) ) return 0;
    fclose(fp);
```

```
    printf("Stage 4 clear!\n");

    // network
    int sd, cd;
    struct sockaddr_in saddr, caddr;
    sd = socket(AF_INET, SOCK_STREAM, 0);
    if(sd == -1){
        printf("socket error, tell admin\n");
        return 0;
    }
    saddr.sin_family = AF_INET;
    saddr.sin_addr.s_addr = INADDR_ANY;
    saddr.sin_port = htons( atoi(argv['C']) );
    if(bind(sd, (struct sockaddr*)&saddr, sizeof(saddr)) < 0){
        printf("bind error, use another port\n");
            return 1;
    }
    listen(sd, 1);
    int c = sizeof(struct sockaddr_in);
    cd = accept(sd, (struct sockaddr *)&caddr, (socklen_t*)&c);
    if(cd < 0){
        printf("accept error, tell admin\n");
        return 0;
    }
    if( recv(cd, buf, 4, 0) != 4 ) return 0;
    if(memcmp(buf, "\xde\xad\xbe\xef", 4)) return 0;
    printf("Stage 5 clear!\n");
```

```
    // here's your flag
    system("/bin/cat flag");
    return 0;
}
```

题目中总共有5个Stage，意味着需要过关，依次通过之后，即可能拿到flag。
先看第一个stage
其代码为

```
    // argv
    if(argc != 100) return 0;
    if(strcmp(argv['A'],"\x00")) return 0;
    if(strcmp(argv['B'],"\x20\x0a\x0d")) return 0;
    printf("Stage 1 clear!\n");
```

首先需要满足的条件为agrc为100，且 'A'（65）个和第'B'(66)个分别是\x00和\x20\x0a\x0d，因此需要构造list，一般默认情况下，argv[0]是"./input"，即所对应的程序名

```
    argv = list('1' * 100 )
    argv[0] = "./input"
    argv[ord('A')] = "\x00"
    argv[ord('B')] = "\x20\x0a\x0d"
```

接着看第二个stage stdio

```
    // stdio
    char buf[4];
```

```
    read(0, buf, 4);
    if(memcmp(buf, "\x00\x0a\x00\xff", 4)) return 0;
    read(2, buf, 4);
        if(memcmp(buf, "\x00\x0a\x02\xff", 4)) return 0;
    printf("Stage 2 clear!\n");
```

第一个memcmp是从stdin中读取数据，与\x00\x0a\x00\xff进行比较，第二个memcmp从stderr中读取数据进行对比。
pwntools中的process有两个参数，分别为stdin和stderr，传入文件对象即可。

```
with open("stdin.txt", "wb") as file:
    file.write("\x00\x0a\x00\xff")
    file.close()
with open("stderr.txt", "wb") as file:
    file.write("\x00\x0a\x02\xff")
    file.close()
```

接着看stage3 env

```
// env
    if(strcmp("\xca\xfe\xba\xbe", getenv("\xde\xad\xbe\xef"))) return 0;
    printf("Stage 3 clear!\n");
```

可以使用process中的env参数，其中的env是字典形式
env = {"\xde\xad\xbe\xef": "\xca\xfe\xba\xbe"}
接着看第四关 file

```
    // file
```

```
    FILE* fp = fopen("\x0a", "r");
    if(!fp) return 0;
    if( fread(buf, 4, 1, fp)!=1 ) return 0;
    if( memcmp(buf, "\x00\x00\x00\x00", 4) ) return 0;
    fclose(fp);
    printf("Stage 4 clear!\n");
```

创建个名字为\x0a的文件，内容为\x00\x00\x00\x00

```
with open("\x0a", "wb") as file:
    file.write("\x00\x00\x00\x00")
    file.close()
```

第五关 network

```
    // network
    int sd, cd;
    struct sockaddr_in saddr, caddr;
    sd = socket(AF_INET, SOCK_STREAM, 0);
    if(sd == -1){
        printf("socket error, tell admin\n");
        return 0;
    }
    saddr.sin_family = AF_INET;
    saddr.sin_addr.s_addr = INADDR_ANY;
    saddr.sin_port = htons( atoi(argv['C']) );
    if(bind(sd, (struct sockaddr*)&saddr, sizeof(saddr)) < 0){
```

```
        printf("bind error, use another port\n");
            return 1;
    }
    listen(sd, 1);
    int c = sizeof(struct sockaddr_in);
    cd = accept(sd, (struct sockaddr *)&caddr, (socklen_t*)&c);
    if(cd < 0){
        printf("accept error, tell admin\n");
        return 0;
    }
    if( recv(cd, buf, 4, 0) != 4 ) return 0;
    if(memcmp(buf, "\xde\xad\xbe\xef", 4)) return 0;
    printf("Stage 5 clear!\n");
```

建立一个socket来接收数据，与 \xde\xad\xbe\xef 进行比较
saddr.sin_family = AF_INET;
saddr.sin_addr.s_addr = INADDR_ANY;
saddr.sin_port = htons(atoi(argv['C']));
指的是需要指定绑定的地址，
INADDR_ANY指的是表示不确定地址，或“所有地址”、“任意地址”，127.0.0.1也包含在内，第二句指定绑定端口，端口号为argv['C']中的内容，需要设置一个不与其他程序冲突的端口号即可。
pwntools中的remote即可实现

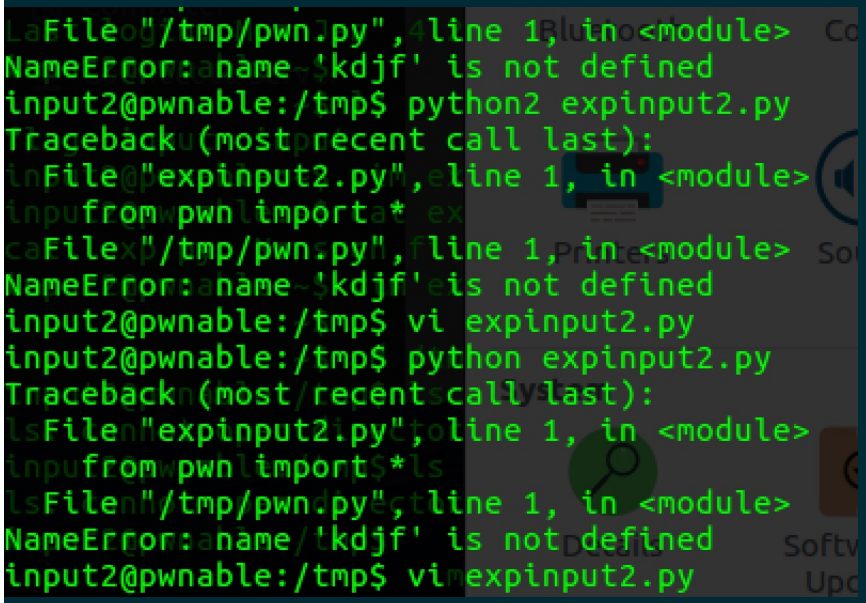
```
r = remote("127.0.0.1", 9999)
r.send("\xde\xad\xbe\xef")
```

因此写出完整的payload

```python
from pwn import *

# stage 1 process
argv = list('1' * 100)
argv[0] = "./input"
argv[ord('A')] = "\x00"
argv[ord('B')] = "\x20\x0a\x0d"

# stage 2 stdio
with open("stdin.txt", "wb") as file:
    file.write("\x00\x0a\x00\xff")
    file.close()
with open("stderr.txt", "wb") as file:
    file.write("\x00\x0a\x02\xff")
    file.close()

# stage 3 env
env = {"\xde\xad\xbe\xef": "\xca\xfe\xba\xbe"}

# stage 4 file
with open("\x0a", "wb") as file:
    file.write("\x00\x00\x00\x00")
    file.close()

# stage 5 network
argv[ord('C')] = "9999"

p = process(argv=argv, env=env, stdin=open("stdin.txt","rb"), stderr=open("stderr.txt","rb"))
```

```
r = remote("127.0.0.1", 9999)
r.send("\xde\xad\xbe\xef")
r.close()

print p.recv()
print p.recv()
```

使用pwntools生成的exp
scp -P 2222 expinput2.py input2@pwnable.kr:/tmp
把exp上传到服务器
/tmp目录下进行代码执行，
ln /home/input/flag flag
服务端做了权限设置，复现失败



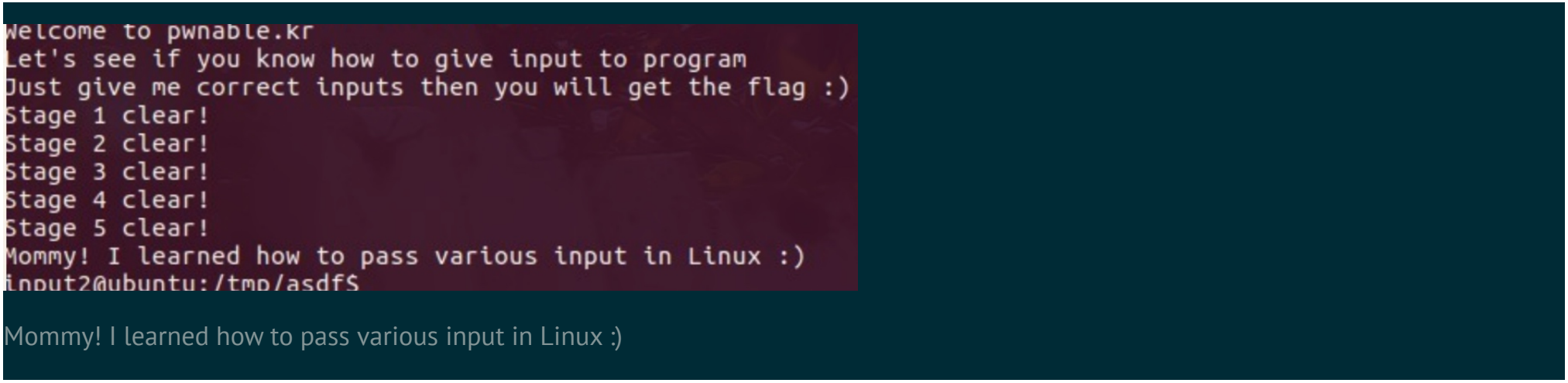
使用c语言编程，

```
#include <stdio.h>
```

```c
#include <unistd.h>
#include <sys/types.h>
#include <stdlib.h>
#include <sys/wait.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>

int main (){
//stage1
    char *argv[101]={"/home/shelldon/Desktop/input"};
    for(int i=1;i<100;i++)argv[i]="A";
    argv[100]=NULL;
    argv['A']="\x00";
    argv['B']="\x20\x0a\x0d";
    argv['C']="55555";


//stage2
 int pipe2stdin[2] = {-1,-1};
int pipe2stderr[2] = {-1,-1};
pid_t childpid;

if ( pipe(pipe2stdin) < 0 || pipe(pipe2stderr) < 0){
    printf("Cannot create the pipe\n");
    return 0;
}
```

```c
if ( ( childpid = fork() ) < 0 ){
    printf("Cannot fork\n");
    return 0;
}

if ( childpid == 0 ){
    /* Child process*/
    close(pipe2stdin[0]); close(pipe2stderr[0]); // Close pipes for reading
    write(pipe2stdin[1],"\x00\x0a\x00\xff",4);
    write(pipe2stderr[1],"\x00\x0a\x02\xff",4);
}
else {
     /* Parent process */
    //sleep(0.1);
    close(pipe2stdin[1]); close(pipe2stderr[1]);   // Close pipes for writing
    dup2(pipe2stdin[0],0); dup2(pipe2stderr[0],2); // Map to stdin and stderr
    close(pipe2stdin[0]); close(pipe2stderr[1]);   // Close write end (the fd has been copied
before)
    //stage3
    char* env[2]={"\xde\xad\xbe\xef=\xca\xfe\xba\xbe",NULL};
    //stage4
    FILE* fp = fopen("\x0a","w");
    fwrite("\x00\x00\x00\x00",4,1,fp);
    fclose(fp);
    execve("/home/shelldon/Desktop/input",argv,env);// Execute the program
```

```
}
//stage5
    sleep(5);
    int sockfd;
    struct sockaddr_in server;
    sockfd = socket(AF_INET,SOCK_STREAM,0);
    if ( sockfd < 0){
     perror("Cannot create the socket");
        exit(1);
    }
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = inet_addr("127.0.0.1");
    server.sin_port = htons(55555);
    if ( connect(sockfd, (struct sockaddr*) &server, sizeof(server)) < 0 ){
     perror("Problem connecting");
     exit(1);
    }
    char buf[4] = "\xde\xad\xbe\xef";
    write(sockfd,buf,4);
    close(sockfd);
    return 0;
}
```



Mommy! I learned how to pass various input in Linux :)