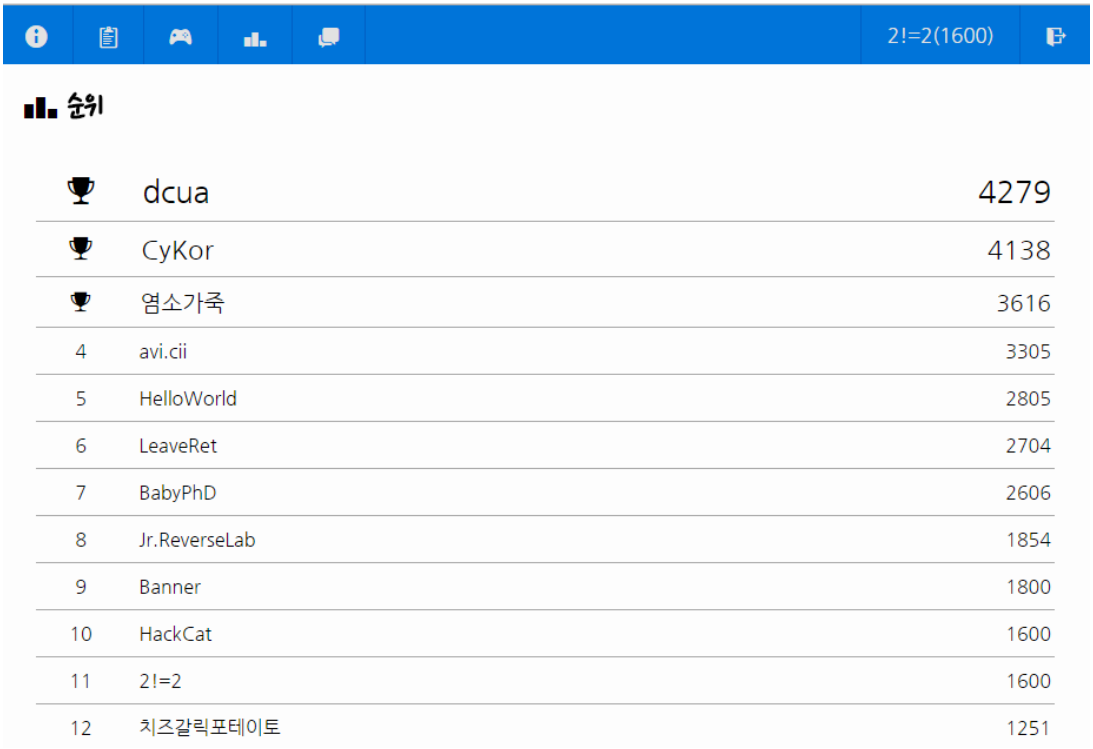
2015 Inc0gnito CTF Write-Up

5kyc1ad(Mini)



2!=2(1600)

순위

	dcua	4279
	CyKor	4138
	염소가죽	3616
4	avi.cii	3305
5	HelloWorld	2805
6	LeaveRet	2704
7	BabyPhD	2606
8	Jr.ReverseLab	1854
9	Banner	1800
10	HackCat	1600
11	2!=2	1600
12	치즈갈릭포테이토	1251

이번 2015 Inc0gnito CTF 에는 후배들과 함께 '2!=2' 라는 팀명으로 참가하였습니다.

총 1600 점으로 11 위를 차지하였습니다.

팀 정보 - 2!=2

아이디: Mini

점수: 1600

팀 코드: cd58701bf6e16a2b

마지막 인증: 2015년 8월 23일 5:23 오후

푼 문제

panic (250)
reversing (150)
Inside Out (150)
OSS (250)
유출 추정 (150)
Anti Hexray (100)
멘붕에 빠진 개발이 (200)
CFT (150)
카카오프렌즈와 문제 풀기 (200)

푼 문제들은 위와 같습니다.

저는 저중에서 OSS(Web), Anti Hexray(Reversing), 카카오프렌즈와 문제 풀기(Web)를 풀었습니다.

[Web] OSS (250pt)

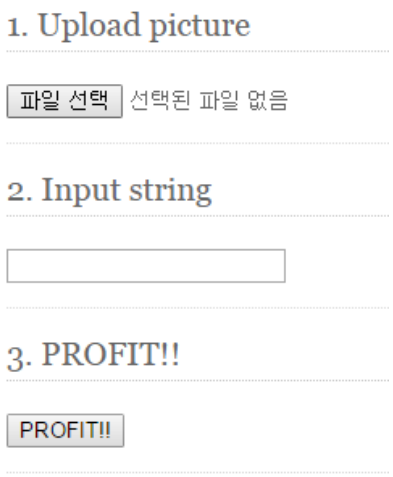
[Web] OSS 250

웹 개발자 김00 씨는 열혈한 아이유의 팬이다. 김씨는 야근때문에 지난달 아이유 콘서트에 참가하지 못해 상사에게 엄청 큰 불만을 품고 있었다. 마침 이번 프로젝트로 그는 웹 페이지의 주요 기능을 제작하게 되는 중책을 맡게 되는데, 그의 표정이 심상치 않다. 그가 만든 취약점을 조기에 발견하여 수정하자!
http://ssh.inc0gnito.com:8912/

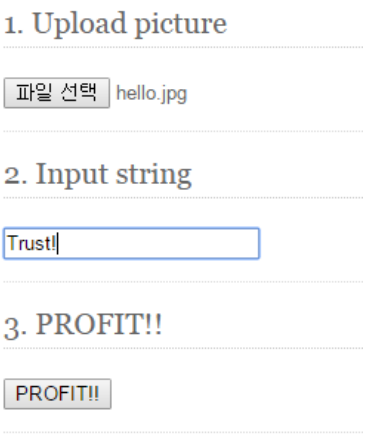
푼 팀: 14/173

아이유의 팬이라는 게 참 낯익습니다.

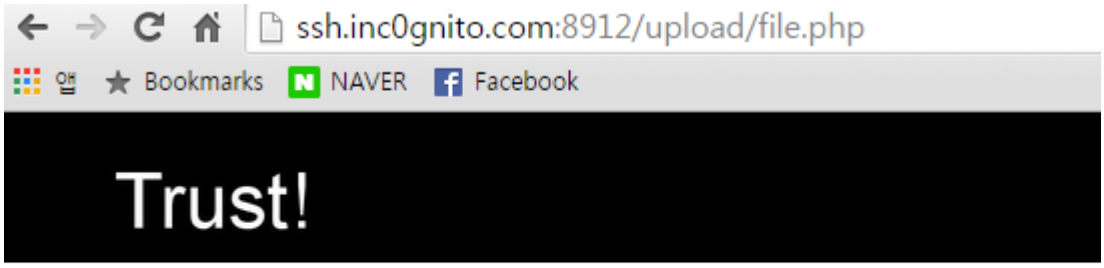
여러 대회에서 많은 문제를 출제하신 분인 듯 합니다.



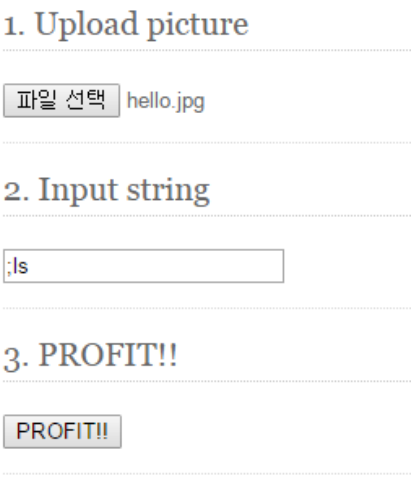
해당 사이트에 들어가면 이렇게 JPG 이미지 파일, 문자열을 넣을 수 있는 칸이 나타납니다.



이렇게 JPG 파일과 문자열을 집어넣으면



입력한 문자열을 해당 JPG 파일에 입력하여 보여줍니다.

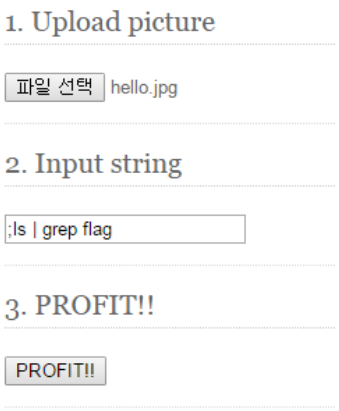


어쩌다 보니 바로 취약점을 찾게 되었는데, 세미콜론을 입력 후 뒤에 명령어를 치게 되면 해당 명령어가 실행된 후 출력된 값이 그대로 이미지 파일에 입력되어 나타납니다.

이런 식으로.

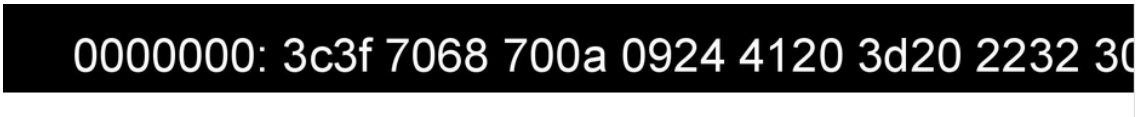
몇 번 시도해본 결과 이미지 파일에 나타나는 입력 값들은 여러 줄이 출력된다고 하면 마지막 한 줄만이 나타납니다. 그것도 너무 길 경우 오른쪽이 조금 잘려서 나타납니다.

가로 1600px 정도가 넣을 수 있는 이미지의 최대 크기이기 때문에 그 이상 늘릴 수도 없어 긴 문자열 등을 출력할 때는 잘릴 가능성을 동반합니다.



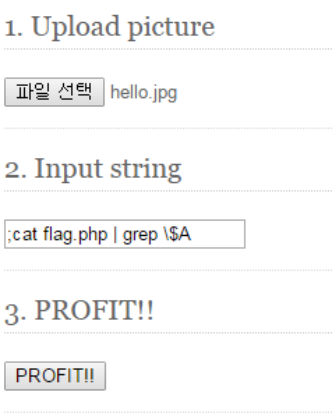
flag 를 찾아야 하므로 grep 으로 걸러내어 flag 라는 문자열을 포함한 파일만을 걸러내었습니다.

이미지 파일에 flag.php 가 나타난 것을 보아 해당 파일이 디렉토리에 존재한다고 볼 수 있습니다.



" ;xxd flag.php | grep 0000000 " 을 String 으로 집어넣으면 xxd 로 출력된 값에서 가장 윗부분을 가져옵니다. 위의 3c3f~ 를 전부 16 진수로 변환하였더니 " <? php ~ " 라는 것을 알 수 있습니다.

첫 줄의 앞부분을 알았으니 이제 cat 와 grep 을 이용하여 데이터를 빼낼 수 있습니다.



이를 계속 활용하여 " ;xxd flag.php | grep 0000010 " 등을 입력하여 얻어 낸 값들을 위와 같이

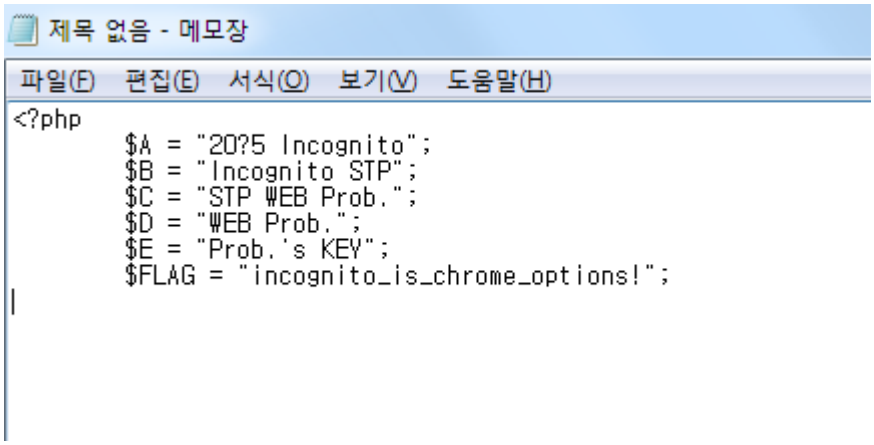
" ;cat flag.php | grep ~ " 으로 해당 열만 가져와서 출력하도록 했습니다.

$A 가 포함된 열을 출력하라고 했더니

이렇게 나타났습니다.

조금 더 하다 보니 한 줄이 변경될 때마다 $A, $B, $C 등으로 변화한다는 것을 알 수 있었고,

이를 토대로 flag 가 나타날 때까지 찾아본 결과



위와 같은 값을 얻게 되었습니다.

flag : incognito_is_chrome_options!

[Web] 카카오프렌즈와 문제 풀기 (200pt)

OSS 와 달리 문제를 풀면서 캡쳐를 하지 못했기 때문에 이미지가 없는 점은 양해를 부탁드립니다.

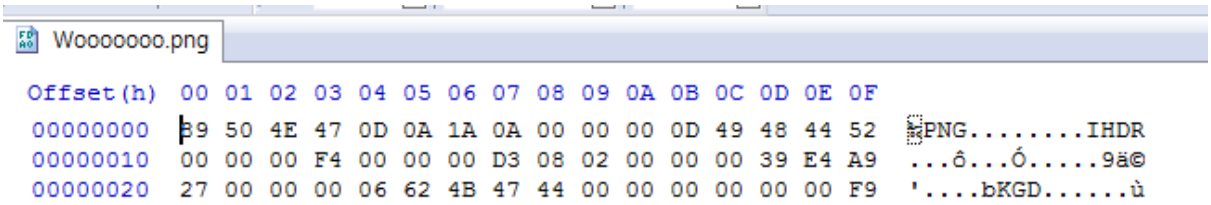
문제에서 주어지는 URL 에 접속하면 ID 란에 guest, PW 란에 'guest' 가 적혀 있는 로그인 폼이 있습니다.

이 부분은 간단한 SQL Injection 으로 PW 에 'AAAAA' or id='admin' 이라는 값을 넣어 주는 것으로 간단히 우회가 가능합니다.

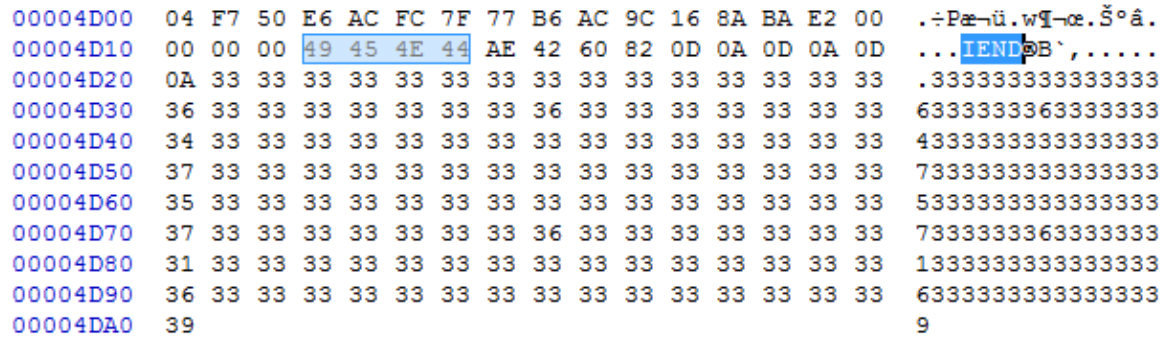
admin 으로 접속하면 Board 라는 곳으로 이동하는데, 여러 개의 글이 있고 그 글들마다 첨부파일이 있어 이미지나 파일 등을 다운받을 수 있습니다.

그 중에서 'Wooooooo.png' 파일을 주는 곳이 있는데, 이 파일은 png 파일이 아니라고 하면서 줍니다.

파일 시그니쳐를 확인해 봤습니다.



첫 부분은 PNG 파일이 맞지만,



끝나는 부분을 확인해 보니 PNG 헤더가 모두 끝난 후에도 이상한 데이터들이 남아있습니다.

여기서 조금 헤메었는데, 잘 보면 0x33 이라는 값은 ASCII 로 바꾸면 '3' 이고, 0x36 은 '6', 0x37 은 '7'이라는 것을 알 수 있습니다. 즉, 간단하게 말해서 Hex 두 자리 중 왼쪽의 3 하나를 지운 값이 곧 ASCII 문자와 같다는 것입니다. 이를 이용해 Hex 를 String 으로 변환해주는 곳을 이용하여 몇 번 돌려서 문자열을 구한 결과

이렇게 muzi 라는 값을 얻을 수 있습니다.

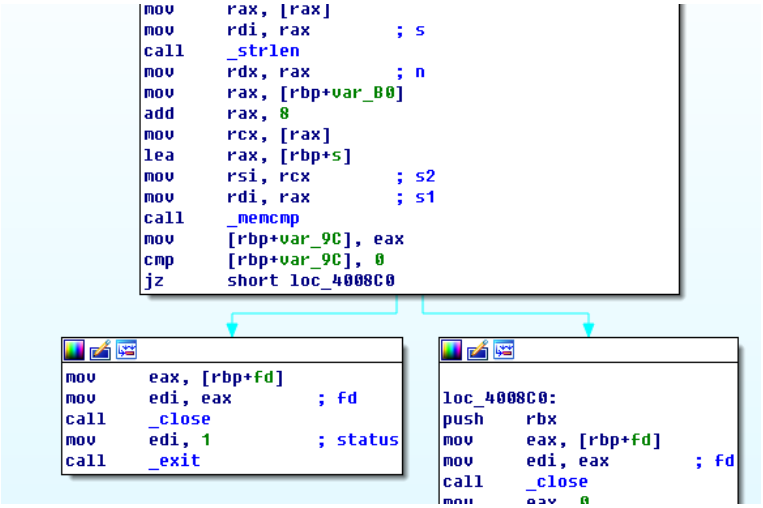
이게 그대로 flag 값입니다.

flag : muzi

[Reversing] Anti Hexray (100pt)

헥스레이를 사용할 수 없는 리버싱 문제인 모양입니다.

솔직히 루틴이 간단해서 안써도 괜찮습니다.



가장 중요한 루틴입니다.

argv 로 입력받은 값을 flag 에서 뽑아낸 flag 값과 memcmp 로 비교하는데, memcmp 특성상 비교할 길이를 지정해 주는데, 그 길이는 argv 의 길이입니다.

즉, 한 글자를 입력하면 한 글자만 비교하고, 두 글자를 입력하면 두 글자를 비교합니다.

틀릴 경우 exit(1)로 비정상 종료 되므로 프로그램 수행 후 "echo $?"를 입력하는 것으로 리턴된 값을 확인하여 입력한 값이 맞는지 틀린지를 확인할 수 있습니다.

이런 식으로 앞부터 한 글자씩 찾아나가면 됩니다.

원래 파이썬으로 자동화 툴을 만드려고 했으나 os.system("echo $?")를 수행시 계속 0 만 나와서 손으로 직접 돌렸습니다.

```
anti_hexray@incognito1:/tmp/mini$ ./anti_hexray "IcEwAll&Inc0gnito"
anti_hexray@incognito1:/tmp/mini$ echo $?
0
anti_hexray@incognito1:/tmp/mini$
```

최종적으로 찾은 값은 위와 같습니다.

ls -al 로 본 파일 크기가 18 바이트였으므로 끝에 NULL 값이 있다는 가정 하에 총 18 바이트가 되어 정확히 일치하는 값입니다.

flag : IcEwAll&Inc0gnito