安全小课堂第113期【二次注入漏洞挖掘】

京东安全应急响应中心 10月9日

Web应用程序已经从上世纪简单的脚本演变成了如今的单页应用。然而，随着Web应用程序的不断复杂化不同类型的安全漏洞也随之而来。其中有一种被称之为二次注入的漏洞，该漏洞是一种在Web应用程序中广泛存在的安全漏洞形式。相对于一次注入漏洞而言，二次注入漏洞更难以被发现，但是它却具有与一次注入攻击漏洞相同的攻击威力。

JSRC **安全小课堂第113期**，邀请到**Cr0w**作为讲师就**二次注入漏洞挖掘**为大家进行分享。同时感谢各位小伙伴们的精彩讨论

什么是二次注入漏洞

京安小妹

Cr0w：

二次SQL注入，也称为二阶SQL注入，其原理和一阶SQL注入（普通SQL注入）相同，均通过精心构造的网络请求（sql语句）获取数据库返回结果，二次注入是利用存储在数据库中的合法内容，追加构造SQL查询语句，达到SQL注入攻击的目的。

二次注入漏洞的危害

京安小妹

Cr0w：

普通SQL注入通过HTTP等网络交互的过程中，很容易被WAF、安全策略、前端JS检测到，虽然利用工具普及，语句的构造方法众多，但大多都在攻击过程中被转义或拦截，甚至被记录了攻击源遭到反渗透。而二次注入攻击，可以一定程度上绕过前端策略的拦截，利用已构造好并存入数据库中的合法内容，通过正规流程/系统去修改无权限的内容，如别人的账号密码。另外，也可以避免一些简单的单引号、双引号、反斜线等转义拦截对构造语句的影响。二次注入攻击，除了在攻击逻辑和寻找利用点有一些难度以外，其攻击成本更低，利用方式更加多变，较难检测。

讲师

二次注入漏洞的攻击原理是什么

京安小妹

Cr0w：

二次SQL注入漏洞原理与常规SQL注入原理一样，通过构造特殊的输入内容传给服务端或数据库达到欺骗服务器执行恶意的SQL命令。因为其利用方式比较独特，需要提前在数据库中插入已经构造好的内容，并利用已有内容进行语句组合提交，方可成功利用，所以称其为二次注入或二阶注入。

讲师

二次注入漏洞实例分析

京安小妹

Cr0w：

因为平时工作比较忙，没来得及去找实例，这部分内容，简单阐述下。

现在大部分的CTF比赛，上面的SQL注入漏洞，都不再是常规的普通注入和绕过利用。

举个例子：

```
$content=addslashes($_POST['title']);
$insert="INSERT INTO A(title,desc)VALUES('$title', '$desc')";
```

很简单的一个插入语句，服务器在插入文章title的过程中使用addsashes进行转义，双引号、单引号、反斜杠、空格都会被添加一个\反斜杠，乍一看是不太好实现SQL注入的，但是如果查询页面是这样

```
$id= addslashes($_GET['id']);
$select_sql="SELECT *FROM A WHERE title='$id'";
```

虽然也被转义，但是在第一步的时候，通过addslashes转义的内容（\）不会插入到数据库中，那我们可以发布一篇文章，title构造成aaa’ union select user(),database()到数据库中，在通过第二段的访问页面找到文章对应的id，访问，便可看到user()和database()信息。

例子：

```
$sql = "UPDATE users set PASSWORD=’$passwd’WHERE USERNAME=’$username’ AND OLD_PASS = ‘$oldpass’";
```

很简单的一句sql，能看出就是一条修改密码的语句，逻辑和内容上都没有问题，那么对于二次注入，问题就从这个$username开始，好多起名字的地方，没有进行严格的昵称校验，导致任何内容都可以写，那么如果系统中存在admin账户，而我们又不知道密码怎么办？新建一个用户，命名为admin’--，admin和admin’--在创建时系统会把它识别为不同用户，可以创建通过，但是在查询的过程中，单引号”’”则为闭合，”--”则为注释，那么查询语句就变成了

```
$sql = "UPDATE users set PASSWORD=’$passwd’WHERE USERNAME=’admin’--’ AND OLD_PASS = ‘$oldpass’";
```

后面的历史密码校验，完全被注释掉，这样，我们就可以通过admin’--用户实现修改admin用户的密码，整个逻辑均属于正常流程。

讲师

二次注入漏洞挖掘思路有哪些

京安小妹

Cr0w：

（1）数据库存储未转义，大多来说，存储到数据库中的东西只是做了表面的转义，实质上库里存储的还是原始状态的内容，在多数读取未校验的情况下，可输入特殊字符并在页面上完整显示的部分，有可能存在二次注入的可能。

（2）强交互部分，SQL注入是需要与数据库联动的，但往往在登录、修改内容、发布内容等功能上忽略了历史数据这一点，所以在遇到有修改账户密码、发布留言、发布内容的部分，大家不妨去试试注册一个test\或者admin'--或者admin\账号，然后在测试回复的内容或者修改的内容是否达到预期。如果与预期不符，那么有可能就存在二次注入，此时在精心构造sql语句即可。

（3）防御绕过，有些地方明显存在异常，但是又做了一些限制，比如年龄，限制了字段为is_numeric，那么我们可以通过转换进制，16进制等方式将其录入，查询时的效果是一样的，遇到其他的限制，跟xss绕过原理一样，大家可以脑洞大开，搞一些数据库可以解析的编码进去。

（4）多录入异常内容，多观察返回结果，凡是与自己录入的内容不一致的地方，就可以去看看是否存在二次注入。

讲师

二次注入漏洞防御手段

京安小妹

Cr0w：

于普通SQL注入防御内容基本一致

过滤、转义。虽然代码规范写的很好，但人为敲代码过程是不可控的，有时候即使转义了，覆盖的接口也不全，即使过滤了，覆盖的内容也不全。

有能力的开发者或者企业，建立一个统一的sql查询api，在api上做统一个防御策略，包括过滤和转义，效果会好很多。这样代码层无论如何调用，通过api的内容完全可以保障其有效性和真实性。

对于普通SQL注入防御的内容就不在阐述了。

讲师

本期JSRC 安全小课堂到此结束。更多内容请期待下期安全小课堂。如果还有你希望出现在安全小课堂内容暂时未出现，也欢迎留言告诉我们。

安全小课堂的往期内容开通了自助查询，点击菜单栏进入“安全小课堂”即可浏览。

简历请发送：cv-security@jd.com

微信公众号：jsrc_team

新浪官方微博：京东安全应急响应中心