众测下的 SQL 注入挖掘

0x01 原理：

sql 注入的原理在这里就不在详细介绍了，我相信大多数师傅对 sql 注入都是有一定的理解，如果对 sql 注入了解不深入的可以详细的去学习一下基础，手工打打靶场，熟悉 sql 函数等。

当然我们在学习 sql 注入的时候我们一定要先去简单的了解各种数据库的特征

比如：**exp()函数来测试注入点，exp(709)正常、exp(710)报错，mssql 和 mysql 适用，oracle 中 exp() 没有固定值，只要能执行我们输入的语句，就可能存在注入**

其余对比有心人可以自己去整理，我相信对你之后在项目中挖掘 sql 注入会有很大的帮助

0x02 测试方法：

在众测的时候可以对 sql 注入挖掘的方法有两种：

1. 被动扫描检查（这一点不建议使用，但是在众测没有说明不能用扫描器的时候可以使用流量较小的扫描）

2. 手工配合 sqlmap 检查：

现在的 web 站点大多数都进行了上 waf 或者进行预编译来防护 sql 注入，但是这两种情况并没有彻底的防护住 sql 注入：

*1： 如果 web 站点有 waf 但可以找出真实 ip，我们可以通过真实 ip 去访问站点，或许这时候我们所访问的真实 ip 就不存在 waf 了进而形成注入

*2： 我们也可以爬取整个 web 站点的接口，因为有可能开发人员在开发的时候，就对一个接口忘记添加 waf 了

*3：在实战中，可控的表名、列名、order by、sort、desc、limit 参数后，不属于用户的输入，也能造成注入，这就是预编译不能完全防范 sql 注入的原因。

常用手法：

单引号报错，双引号正常(对比返回长度)可能存在注入，然后判断数据库类型，套用各数据库语句函数，证明有注，之后可以放入 sqlmap 中跑出注入或者自己写脚本来跑

注入存在点：

1、常见的参数传递，像 id=1(数字型)，id=x(字符串型)、查询框、登录框

2、cookies、http 头、user-agent、ip，也有可能存在注入，不要放过数据包的每个部分。

3、不明显地方，字段排序、表名、字段名可控也能注入。

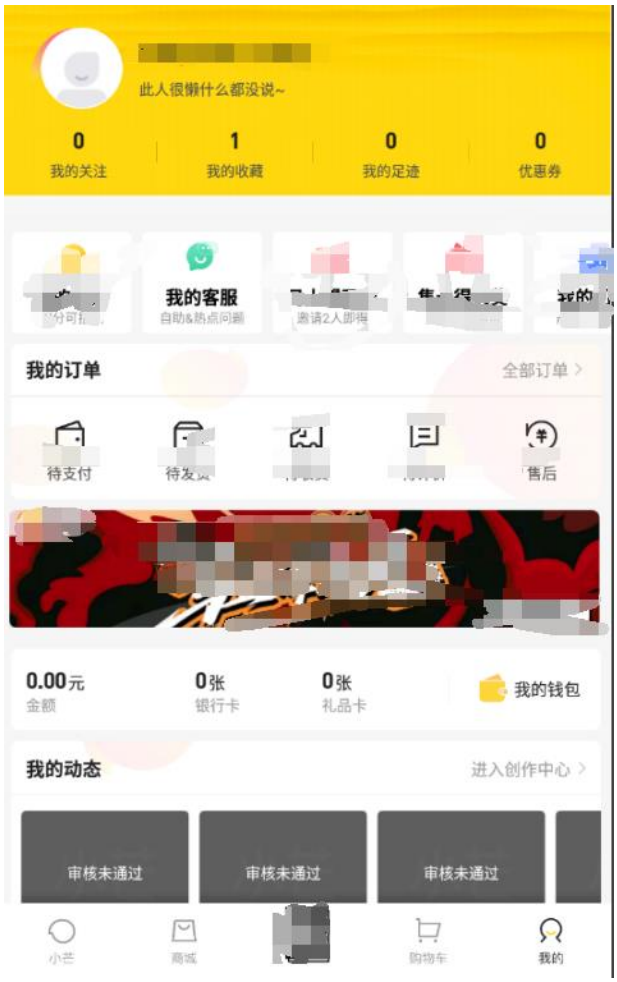
例如：order by id、sort=id、desc、asc、limit 后注入，可以用逗号来闭合，后面接上注入语句，达到注入的结果。

当然我们此次的案例是人工服务处存在的 sql 注入。

0x03 案例：

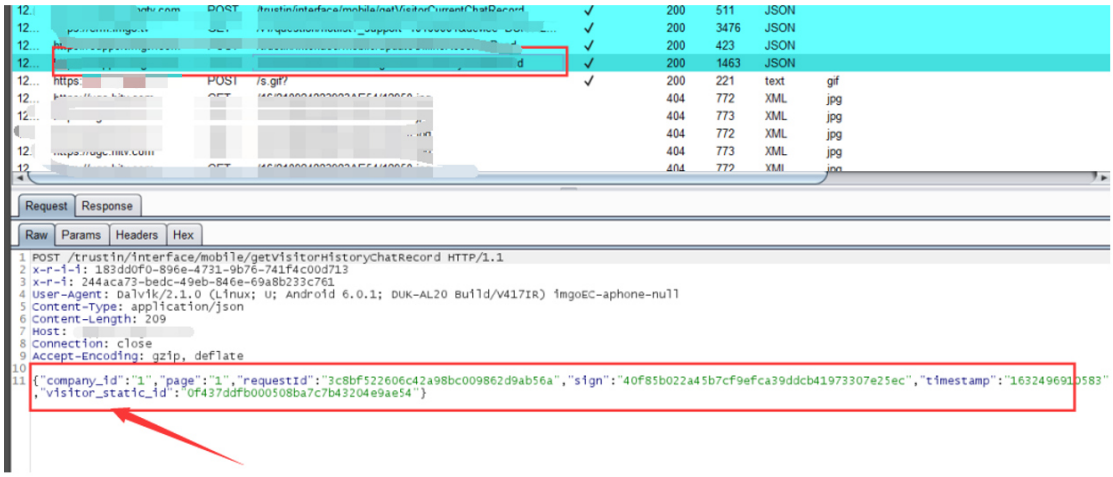
在某安的一次项目中，感觉 web 上的资产确实太少了，而且很多人都在挖掘于是将资产转移到了 app 上面：

当我们下好 app 后，就如下图一样没有什么特别



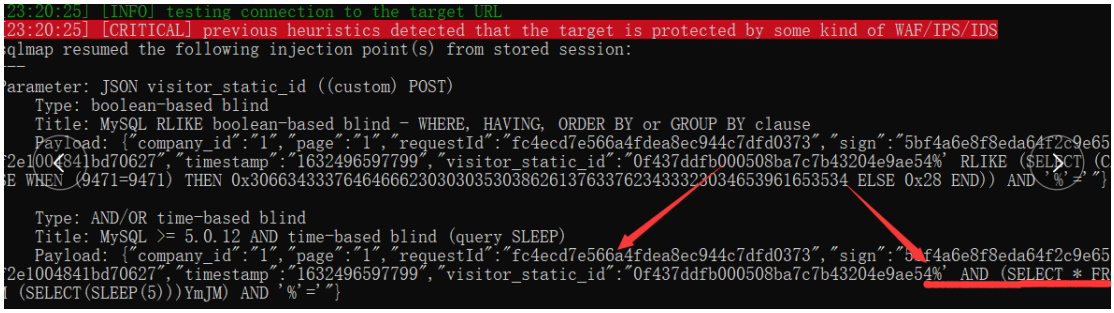
（我个人挖掘 app 的时候喜欢将所有功能点都点一下，然后回去看历史记录）

但是在我把所有功能点都尝试了一次，我发现一个接口是 post 传参，但是在格式中存在 xx_xx_id 等字眼，于是进行单引号测试，没想到成功报错，闭合后数据包返回正常（这里的参数很奇怪，在测试中不要放过任何一个参数，说不定这个参数就有问题）



我相信很多人都只会去测试第一个 id，后面的 id 可能看都不会看一眼，但是在这些出现问题的参数就是最后一个，简单的单引号就可以测试出来。

在确定某参数存在注入后，我就放入 sqlmap 中自动跑出结果即可：



通过 sqlmap 跑出的参数也可以看出是最后一个参数存在注入，当时以为整个站点都会存在，于是各个都跑了一次，最后才发现只有我的客服这一个点存在注入，所以当网站有我的客服这一个点的时候可以进行注入测试。

最后也是获取高危赏金 4k。