

XD安全 学习笔记 | 逻辑漏洞

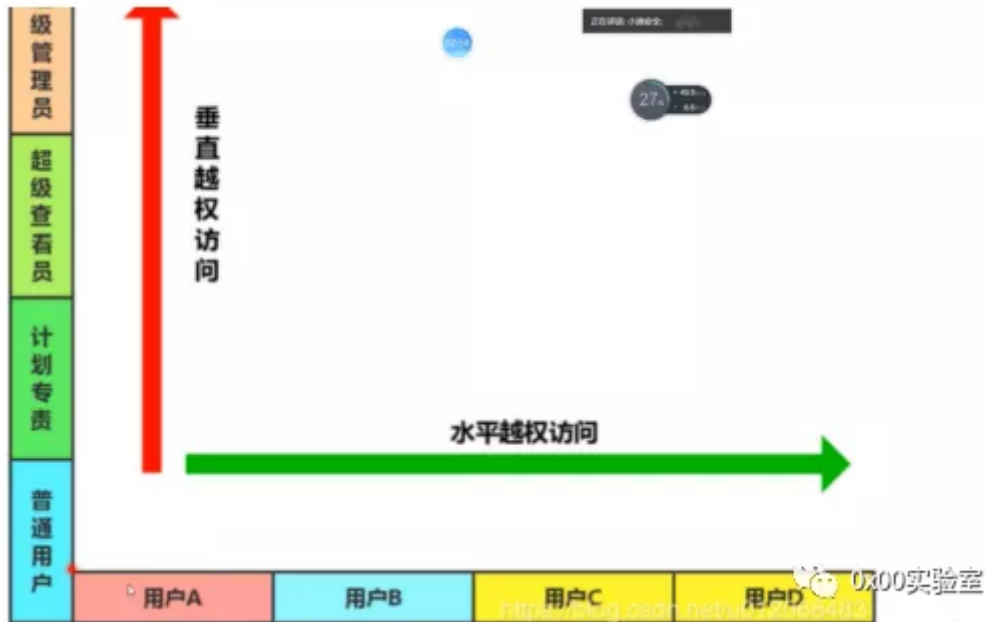
原创 Flyable 0x00实验室 1周前

声明

作者：团队成员-Flyable 【无名安全团队】 如需转载本实验室的文章，请标明来源即可。文章仅学习安全技术使用，切记请勿做它用，产生的后果与本公众号无关。

day33-36天 B站小迪课程的学习笔记。

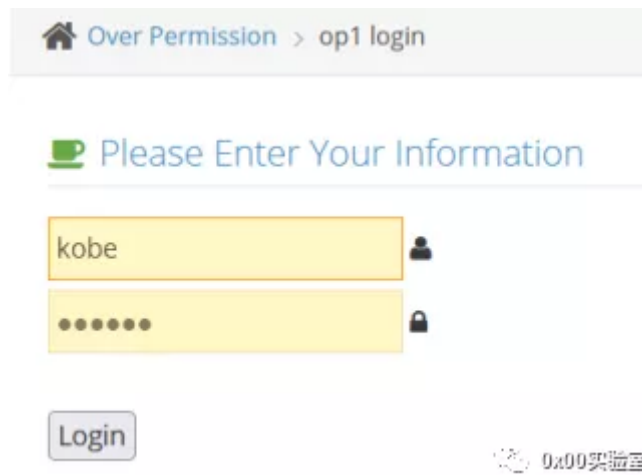
越权分为：水平越权和垂直越权（作用更大），未授权访问



- 1 水平越权：通过更换某个ID之类的身份标识，从而使A账号获取修改B账号数据；
- 2 垂直越权：使用低权限身份的账号，发送高权限账号才能有的请求，获取其更高权限的操作；
- 3 通过删除请求中的认证信息后重放该请求，依旧可以访问或者完成操作

水平越权

pikachu靶场中逻辑水平越权登录kobe相关信息，登录，查看个人信息时对其进行抓包，



查看个人信息时对其进行抓包，可以得到

```
GET /pikachu-master/vul/overpermission/op1/op1_mem.php?username=kobe&submit=%E7%82%B9%E5%B7%B8%E6%9F%A5%E7%9C%8B%E4%B8%AA%E4%BA%BA%E4%BF%A1%E6%81 HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:90.0) Gecko/20100101 Firefox/90.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://127.0.0.1/pikachu-master/vul/overpermission/op1/op1_mem.php
Cookie: PHPSESSID=2ba250640bd8d32b068862231e073a69
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
```

将username更改为lucy，便可得到lucy的相关信息

hello,lucy,你的具体信息如下:

姓名:lucy

性别:girl

手机:12345678922

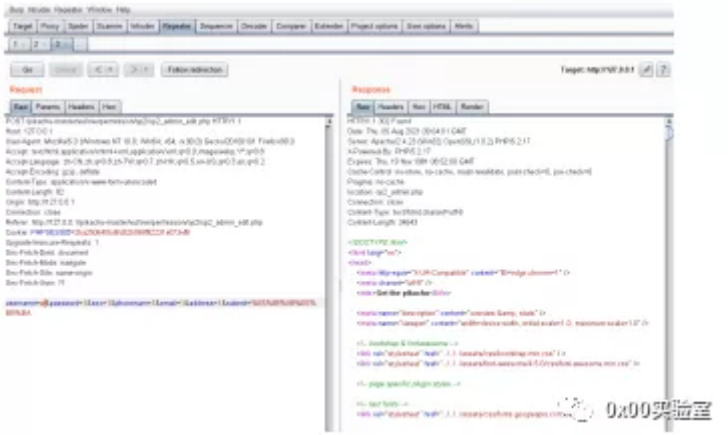
住址:usa

邮箱:lucy@pikachu.com

垂直越权

在登录admin(较高级的管理员)的情况下，对其操作进行抓包，在登录pikachu（低一级用户）时，就可以将较高级的包调出来，更改cookiePHPSESSID=的值，可以条件：需要有admin的数据包

更改相关的user值，便可以对其进行添加



条件：可以抓到admin的数据包

- 1、普通用户前端有操作界面可以抓取数据包
- 2、盲猜
- 3、通过网站源码本地搭建自己去获取

墨者靶场

抓两个数据包，看到有个包的card_id的值，我们考虑用burp suite的thunder功能对其进行爆破，排序相关的length对较大的值进行查看，或者查看图片源地址，可以看到马春生的相关caid_id账号，在该数据包中查看账号和密码（可能需要解密），登录，得到其flag;

```
GET /json.php?card_id=20128880322 HTTP/1.1
```

原理：

- 1、前端问题，界面，判断用户等级之后可选显示，直接根据用户的usertype值判断级别信息。

```
id,username,password,usertype

1,admin,123456,1

2,xiaodi,11111,2
```

登录用户admin或xiaodi时，代码是如何验证这个级别？（usertype判断）

如果在访问网站数据包中有传输用户的编号、用户组编号或类型编号的时候，那么尝试对这个值进行修改，就是测试越权漏洞的基本。

secsan-authcheck插件

检测：

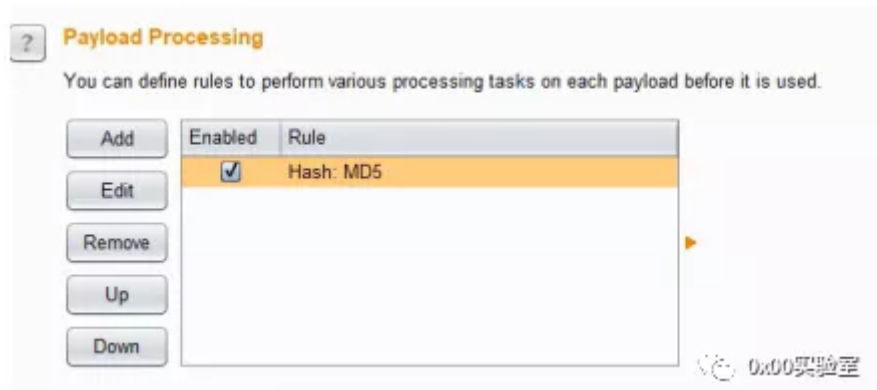
小米范越权漏洞检测工具、burpsuite的anthz插件

#修复防御方案

- 1. 前后端同时对用户输入信息进行校验，双重验证机制
- 2. 调用功能前验证用户是否有权限调用相关功能
- 3. 执行关键操作前必须验证用户身份，验证用户是否具备操作数据的权限
- 4. 直接对象引用的加密资源ID，防止攻击者枚举ID，敏感数据特殊化处理
- 5. 永远不要相信来自用户的输入，对于可控参数进行严格的检查与过滤

浏览器插件authcheck

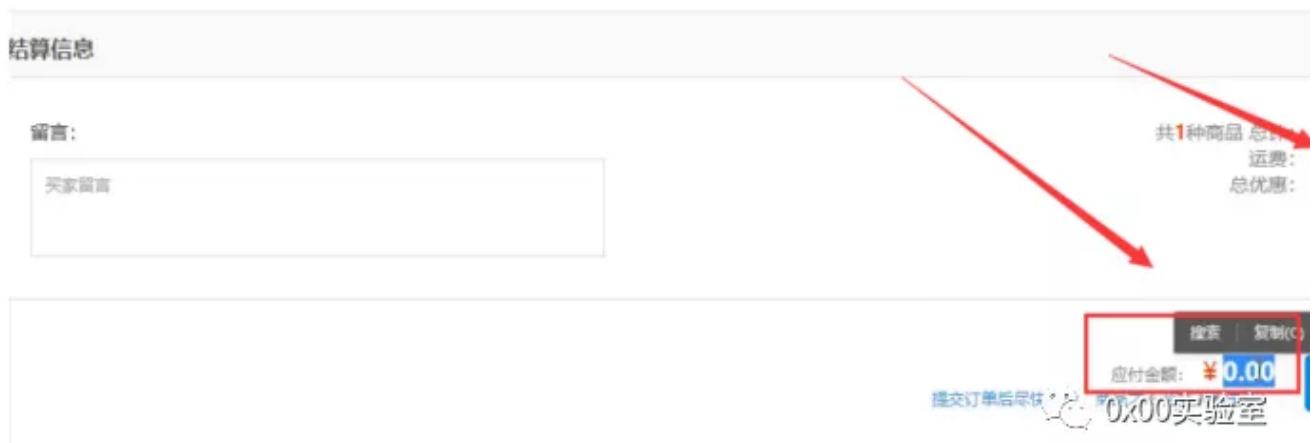
登录应用功能安全问题Http和https协议抓取Http抓包，密码一般是明文传输，也有加密传输，确定其加密方式，进而对其及进行爆破，MD5加密结果一般都是32位对密码进行加密之后爆破。



https抓包，密码一般是密钥传输cookie脆弱性，
代码审计，比如有的只需要验证cookie有价值进行，抓包，给cookie赋一个值，便可成功绕过cookie验证

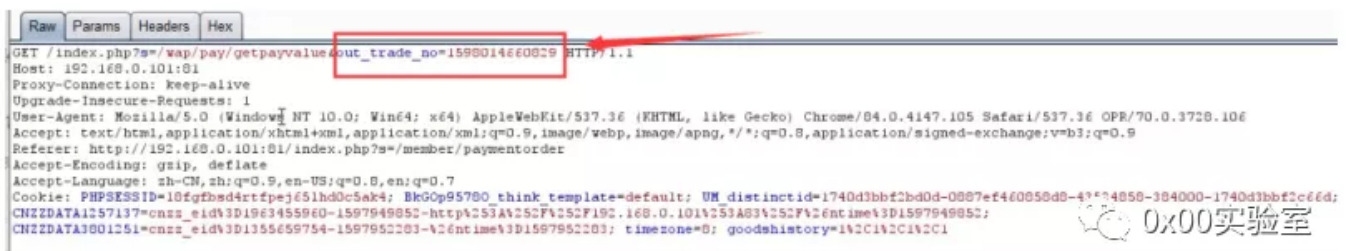
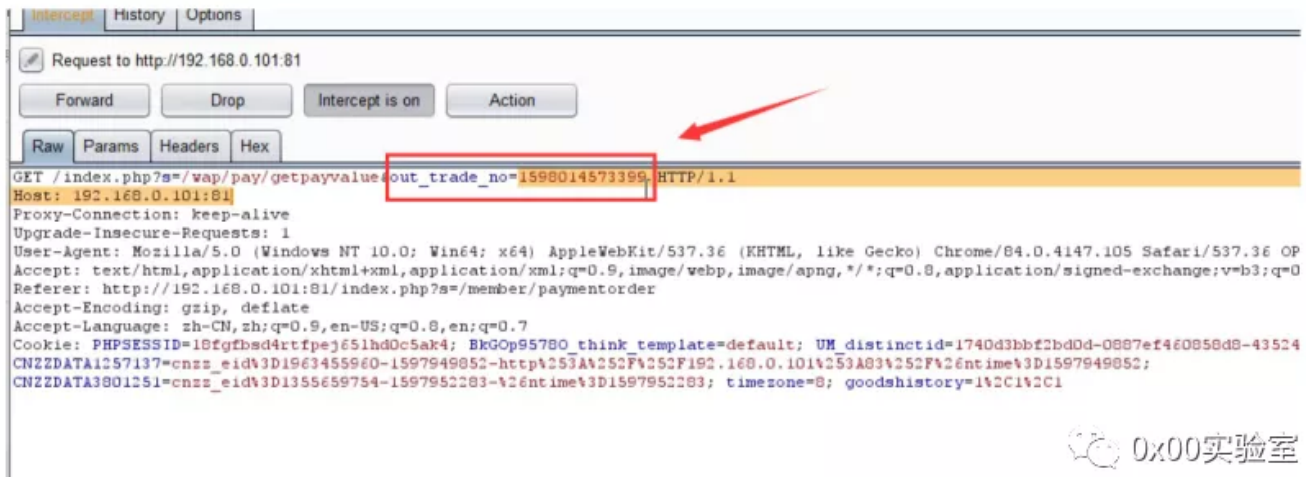
3 #常见篡改参数:
3 商品编号ID, 购买价格, 购买数量, 支付方式, 订单号, 支付状态等

1. 抓包，修改相关值（ID,price,num,statu等），数量可以改为-1



或者是，先抓取订单编号为10元的数据包，然后购买加个1000元的东西时，将数据包得订单编号更改为10元的，然后只需支付10元

2、或者抓包修改相关商品的信息，id值和name值。



3、修改支付状态

找回重置机制：客户端回显，response状态值，验证码爆破（验证码次数，时间），找回流程绕过等接口
调用乱用：短信轰炸，来电轰炸

墨者靶场密码重置-验证码套用-靶场

1 <https://www.mozhe.cn/bug/detail/K2sxTTVYawNncUE1cTdyNXIyTk1Hdz09bw96aGUmozhe>

关于获取某系统帐号权限的说明

关于获取某系统帐号权限的说明

通过“朝阳群众”的举报，证实手机号“17101304128”在系统平台从事非法集资、诈骗活动。

请重置“17101304128”登录密码，以便登录获取完整的数字证据，[点击进入重置密码](#)

0x00实验室

正常情况下，手机验证是第一个页面输入手机号，验证码，第二个页面重置密码，而该题目中，手机验证码和重置密码在同一个地方，步骤：先用要重置密码的手机获取验证码（验证码得不到，不用管），



然后用已经注册的且在身边的手机号获得验证码输入,

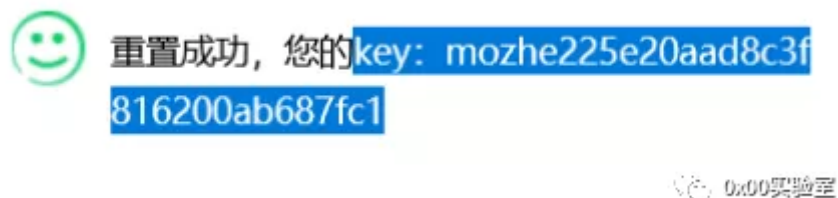


然后抓包，将之前那个号码换成要重置密码的手机号

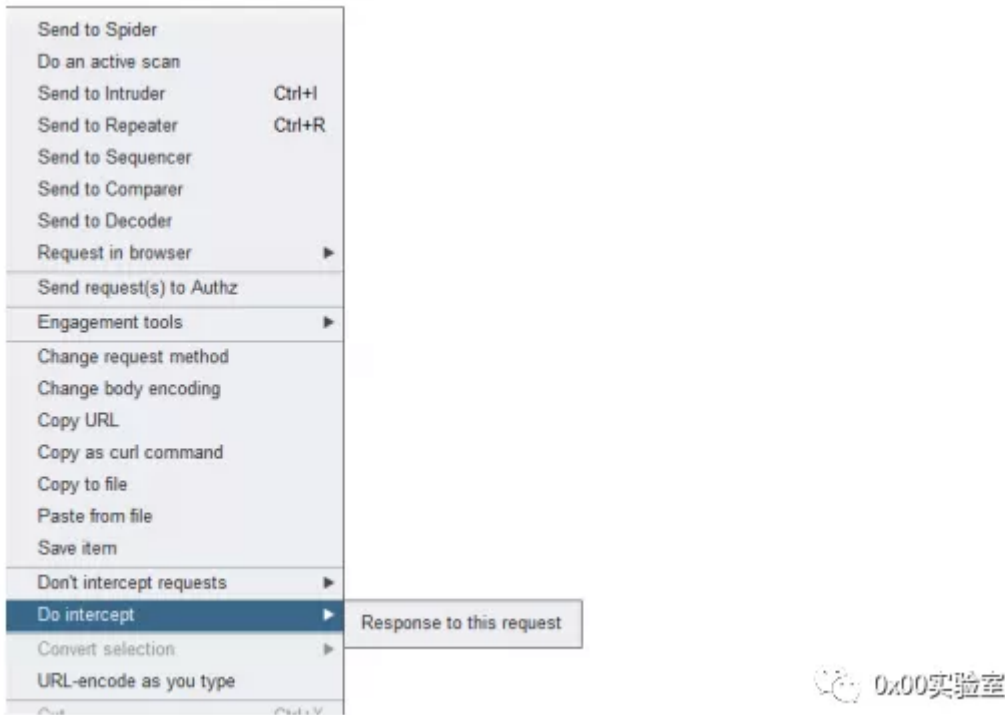
```
Connection: close
Referer: http://219.153.49.228:49081/password_reset.php
Cookie: PHPSESSID=la682nl2rpq9be37rk54isn167

mobile=17101304128&pwd=123456&v_code=gkv&s_code=a31ac5
```

成功获得flag

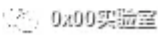


手机邮箱验证码逻辑-客户端回显-实例抓包，在数据包里面，直接可以看到验证码随便输一个验证码，抓包，之后全程接管，将状态码进行更改，例如将3改为1，以当前的回复值修改就有意义，若是以服务器来验证的话，就不行



哈哈哈哈哈，迪哥的手机号

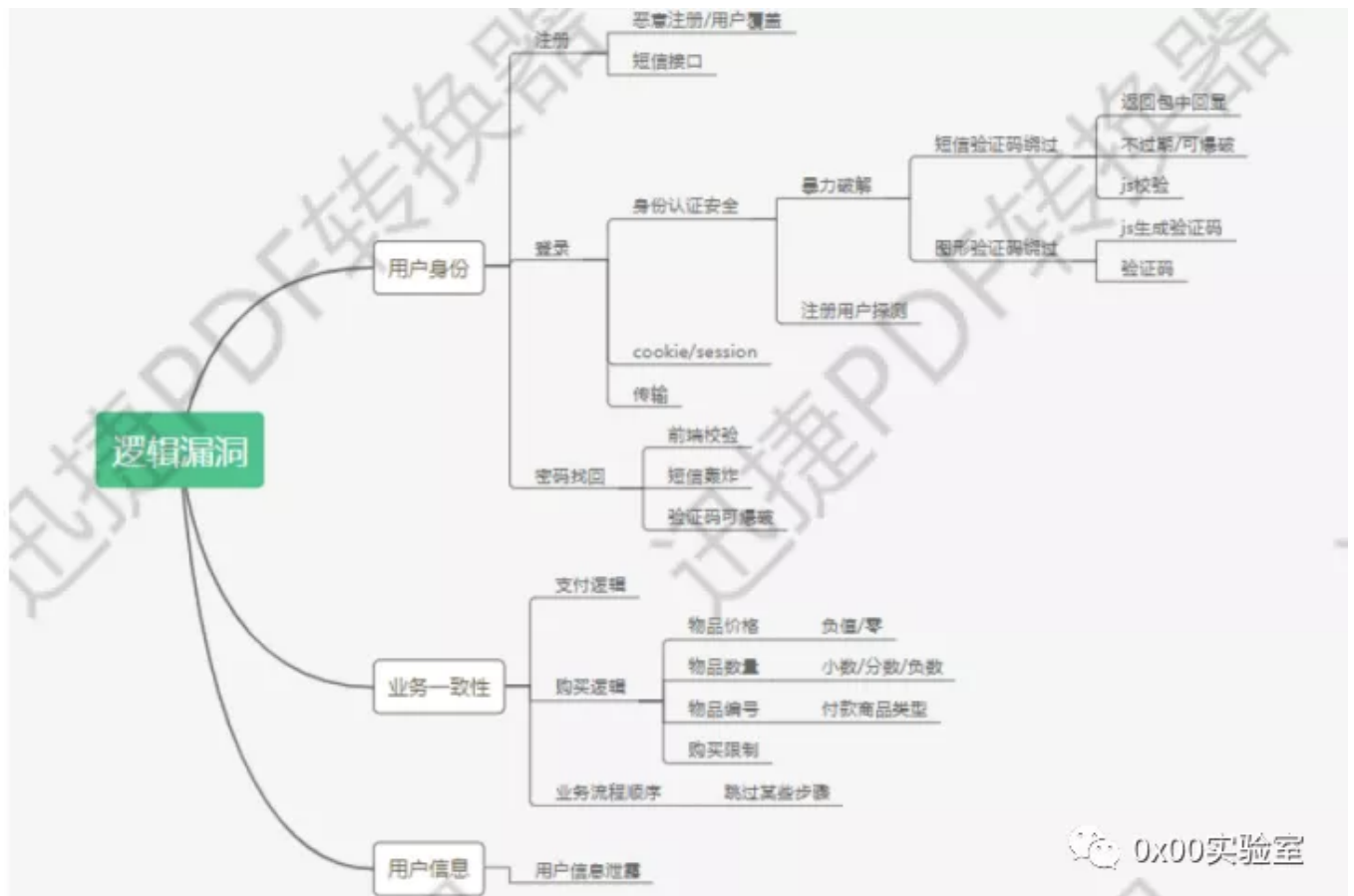
```
{"code":200,"data":"13554365566","msg":""}
```



验证安全：

1、Token(爆破、回显、固定)，回显在浏览器前端可以看见，固定就是可以多次重复使用2、验证码（爆破、识别、复用、回显、绕过）Token是服务端生成的一串字符串，以作客户端进行请求的一个令牌，当第一次登录后，服务器生成一个Token便将此Token返回给客户端，以后客户端只需带上这个Token前来请求数据即可，无需再次带上用户名和密码

可以把页面回显中的Token值替换爆破中的Token值就可以绕过Token



#验证码安全

分类：图片，手机或邮箱，语音，视频，操作等

原理：验证生成或验证过程中的逻辑问题

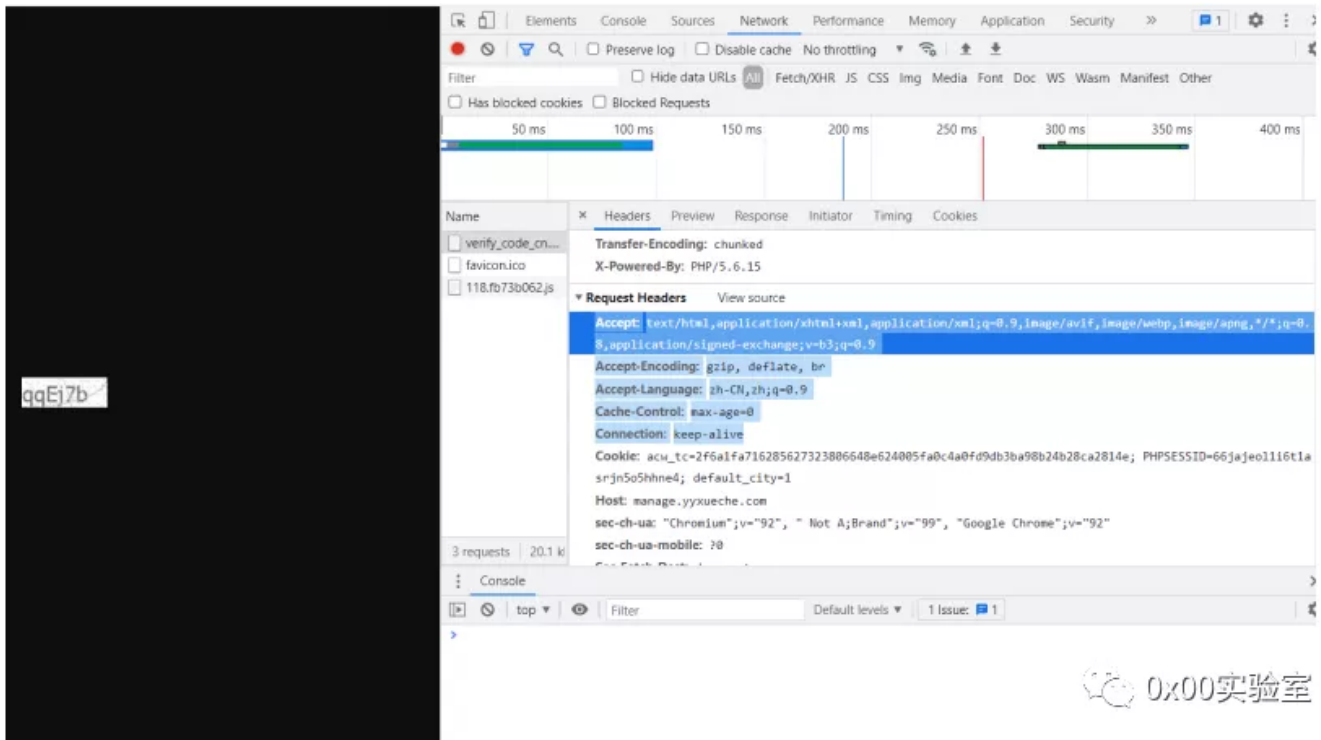
危害：账户权限泄漏，短信轰炸，遍历，任意用户操作等

漏洞：客户端回显(已讲)，验证码复用，验证码爆破(已讲)，绕过等

1 <https://manage.yyxueche.com//panel/login.php>



复制图片地址



验证码地址，为验证码图片的链接地址



爆破防范措施:

1. 如验证码输入3次就重新获取
2. 每输入一下变换一次
3. 使用特殊字符集
4. 特殊的验证码如图形，滑动，判断图形等

使用burp插件captcha-killer识别图片验证码

漏洞产生的原因

通常情况下，一个 Web 程序功能流程是登录 - 提交请求 - 验证权限 - 数据库查询 - 返回结果。如果验证权限不足，便会导致越权。常见的程序都会认为通过登录后即可验证用户的身份，从而不会做下一步验证，最后导致越权。

- 1 隐藏URL
- 2 直接对象引用
- 3 多阶段功能
- 4 静态文件
- 5 平台配置错误
- 6 修复方案
- 7 前后端同时对用户输入信息进行校验，双重验证机制
- 8 调用功能前验证用户是否有权限调用相关功能
- 9 执行关键操作前必须验证用户身份，验证用户是否具备操作数据的权限
- 10 直接对象引用的加密资源ID，防止攻击者枚举ID，敏感数据特殊化处理
- 11 永远不要相信来自用户的输入，对于可控参数进行严格的检查与过滤

burpsuite中的intruder模式中各个选项的具体含义以及遍历方式

target选项卡

设置攻击目标，可以通过proxy发送；

Positions选项卡

指定需要暴力破解的参数并设置成变量，同时选择攻击模式：

Sniper:狙击手

设置一个payload，先将第一个变量使用字典进行测试，然后在将第二个变量使用字典进行测试；

Battering ram:冲撞车


设置一个payload，所有的变量一起用字典内容被替换，然后一起尝试；

Ptichfork:草叉型

每个变量设置一个payload，分别使用对应的字典对变量进行同时替换；

Cluster bomb:焦束炸弹

需要为每个变量设置一个payload，分别使用字典内容组合对变量进行替换；

 0x00实验室

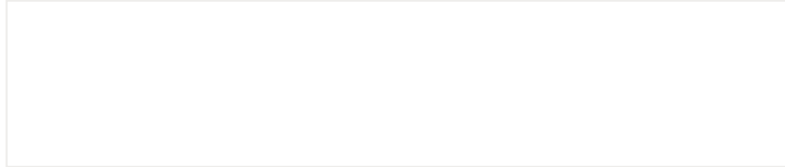
1、**sniper**会先对第一个变量进行字典遍历（第二个变量还是抓包之前输入的那个），然后对第二个变量进行字典遍历；例如之前的输入第二个为q。（只有一个payloadset值为1）

```
username=admin&password=q&submit=Login
```

```
username=q&password=test&submit=Login
```

0x00实验室

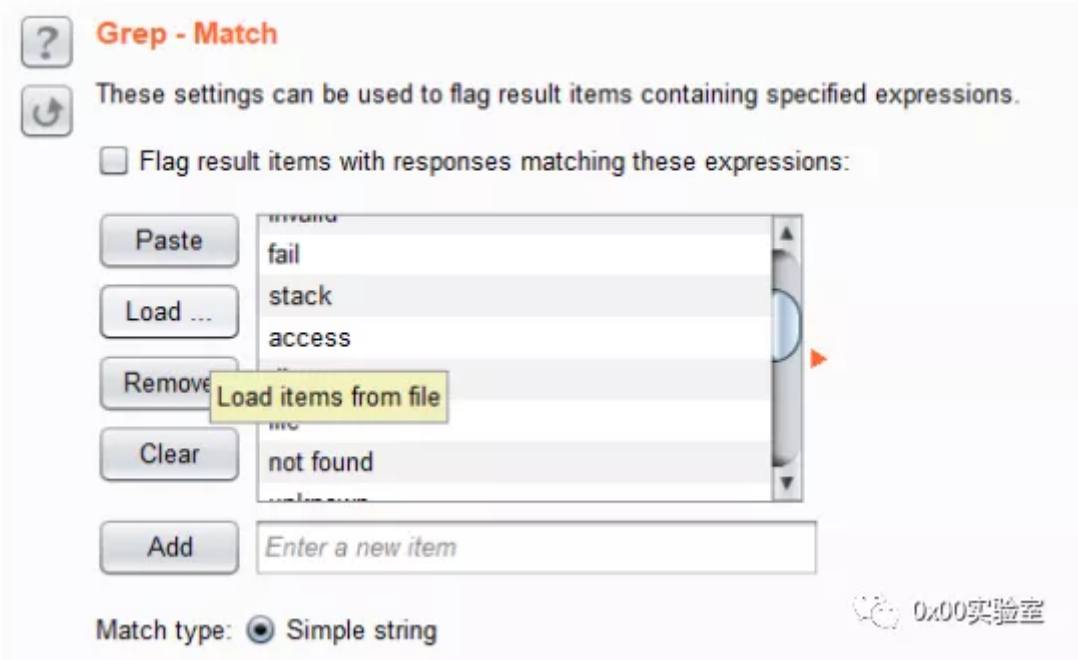
2、**batteringram**会将两个变量同时进行替换，替换同一个字典的值（只有一个payloadset值为1）



3、**pitchfork**，设置几个变量就会有几个payloadset值，会利用第一个字典里面的第一个值对第一个变量进行替换，会利用第二个字典里面的第一个值对第二个变量进行替换，利用第一个字典里面的第二个值对第一个变量进行替换，利用第二个字典里面的第二个值对第二个变量进行替换，利用第一个字典里面的第三个值对第一个变量进行替换，利用第二个字典里面的第三个值对第二个变量进行替换，假如两个字典相同，就会产生下面的结果（一一相对）

Request	Payload1	Payload2	Status	Error	Timeout	Length	Comm
0			200	<input type="checkbox"/>	<input type="checkbox"/>	35076	
1	admin	admin	200	<input type="checkbox"/>	<input type="checkbox"/>	35076	
2	pikachu	123456	200	<input type="checkbox"/>	<input type="checkbox"/>	35076	
3	test	pikachu	200	<input type="checkbox"/>	<input type="checkbox"/>	35076	
4	abc123	000000	200	<input type="checkbox"/>	<input type="checkbox"/>	35076	
5	123456	test	200	<input type="checkbox"/>	<input type="checkbox"/>	35076	
6	000000	abc123	200	<input type="checkbox"/>	<input type="checkbox"/>	35076	

4、**cluster bomb**会对两个字典进行所有组合（使用较多）option中的Grep-Match用来确定那种状态是成功的，哪一种是失败的



flag出错误的内容

