# java代码审计之租车系统

> 这是java代码审计的第一篇文章，初次学习，记录了比较详细的过程。

## 环境搭建

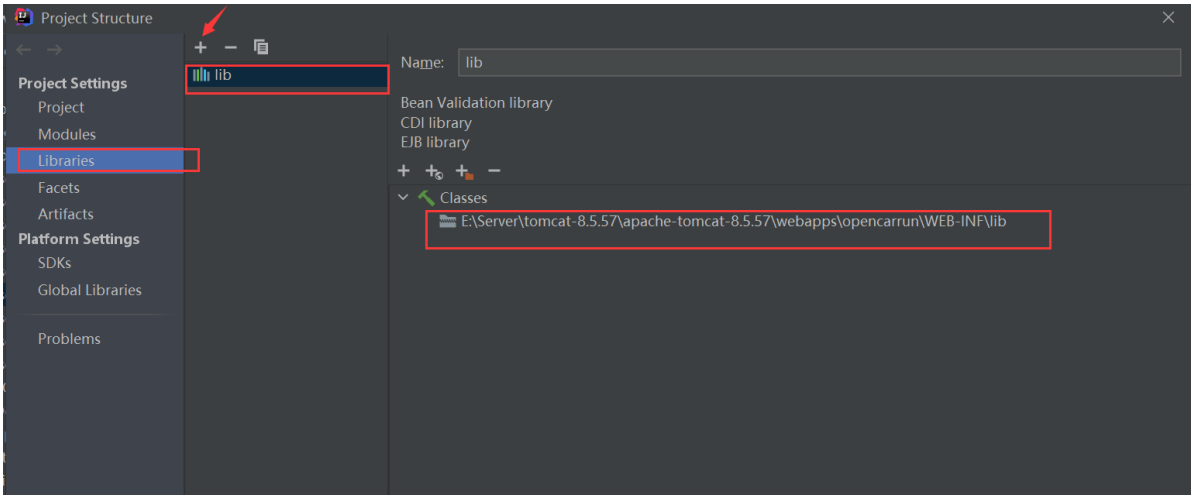
1.下载源代码 如文件中的**zxzcxt.zip**

2.阅读源代码中的使用说明书就ok，注意的是，将sql导入mysql的过程中需要先创建databases.

```
CREATE DATABASE `admin` DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci;
```

3.需要修改源代码中的配置文件，`opencarrun\WEB-INF\classes\db.properties`需要databases和用户名密码。

> 还有一个问题。因为不是自己创建的项目直接用idea打开是字节码，而且跟不走。这里是自己的解决方法，把源代码中的class文件全部copy出来，使用jd-gui.exe反编译成java文件，在通过idea打开并且加载lib文件中的全部内容




## 危险函数

```
密码硬编码、密码明文存储：password、pass、jdbc
```

```
XSS：getParamter、<%=、param.
```

```
SQL 注入：Select、Dao 、from 、delete 、update、insert
```

```
任意文件下载：download 、fileName 、filePath、write、getFile、getWriter
```

```
文件上传：Upload、write、fileName 、filePath
```

```
任意文件删除：delete 、deleteFile、fileName、filePath
```

```
命令注入：getRuntime、exec、cmd、shell
```

缓冲区溢出：strcpy,strcat,scanf,memcpy,memmoGetc(),fgetc(),getchar;read,printf

XML 注入：DocumentBuilder、XMLStreamReader、SAXBuilder、SAXParser
SAXReader 、XMLReader
SAXSource 、TransformerFactory 、SAXTransformerFactory 、
SchemaFactory

反序列化漏洞：反序列化操作一般在导入模版文件、网络通信、数据传输、日志格式化存储、对象数据落磁盘或DB 存储等业务场景,在代码审计时可重点关注一些反序列化操作函数并判断输入是否

ObjectInputStream.readObject
、ObjectInputStream.readUnshared、XMLDecoder.readObject
Yaml.load 、 XStream.fromXML 、 ObjectMapper.readValue 、
JSON.parseObject

日志记录敏感信息：log.info logger.info

URL跳转：sendRedirect、setHeader、forward

敏感信息泄露及错误处理：Getmessage、exception

不安全组件暴露:AndroidManifest.xml
通过查看配置文件 AndroidManifest.xml,查看<inter-filter>属性有没有配置 false

# 短信轰炸

```
    public void getTelCode(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException, DocumentException {
        response.setContentType("text/html;charset=UTF-8");
        request.setCharacterEncoding("UTF-8");
        HttpSession session = request.getSession(true);
        PrintWriter out = response.getWriter();
        String tel = request.getParameter("tel");
        String json = "";
        SendMessageInfo smi = new SendMessageInfo();
        String code = smi.sendMessageInfo(tel, 1,
StringUtil.getRandomString(4));
        if (code.length() < 14) {
            json = "{\"tip\":\"验证码已经发送到你的手机\",\"status\":200}";
            session.removeAttribute("telcode");
            session.setAttribute("telcode", code);
        } else {
            json = "{\"tip\":\"验证码发送失败，您没有购买短信接口，请联系QQ：1919594905
购买短信\",\"status\":200}";
        }

        out.print(json);
    }
```

```java
public void getTelCode(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOExce
    response.setContentType("text/html;charset=UTF-8");
    request.setCharacterEncoding("UTF-8");
    HttpSession session = request.getSession(true);
    PrintWriter out = response.getWriter();
    String tel = request.getParameter("tel");
    String json = "";
    SendMessageInfo smi = new SendMessageInfo();
    String code = smi.sendMessageInfo(tel, 1, StringUtil.getRandomString(4));
    if (code.length() < 14) {
        json = "{\"tip\":\"验证码已经发送到你的手机\",\"status\":200}";
        session.removeAttribute("telcode");
        session.setAttribute("telcode", code);
    } else {
        json = "{\"tip\":\"验证码发送失败，您没有购买短信接口，请联系QQ：1919594905购买短信\",\"status\":200}";
    }

    out.print(json);
}
```

并没有对介绍的tel做验证

可以发好多个

```
POST /opencarrun/front?tag=getTelCode HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:88.0)
Gecko/20100101 Firefox/88.0
Accept: application/json, text/javascript, */*; q=0.01
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
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 14
Origin: http://localhost:8080
Connection: close
Referer: http://localhost:8080/opencarrun/goPetition/10.html
Cookie: JSESSIONID=9B520B2C3854F46A8D473BED17330332
DNT: 1
Sec-GPC: 1

tel=1388888888
```

```
HTTP/1.1 200
Content-Type: text/html;charset=UTF-8
Content-Length: 59
Date: Tue, 01 Jun 2021 10:50:26 GMT
Connection: close

{"tip":"验证码已经发送到你的手机","status":200}
```

# 存储型xss

用户后台修改用户名字的地方。


我们看一些后端代码。

```java
public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    HttpSession session = request.getSession( b: true);
    ShopService ss = new ShopService();
    Locale loc = new Locale( language: "zh", country: "CN");
    ResourceBundle rb = ResourceBundle.getBundle( baseName: "messages", loc);
    String adminTip = rb.getString( key: "adminTip");
    String json = "";
    String name = request.getParameter( s: "name");
    String email = request.getParameter( s: "email");
    String tel = request.getParameter( s: "tel");
    Object ordinary_user = session.getAttribute( s: "ordinary_user");
    if (ordinary_user != null) {
        String flag = "";
        User user = (User)ordinary_user;
        if (name.trim().length() < 2) {
            json = "{\"tip\":\"" + rb.getString( key: "modify") + rb.getString( key: "failure") + ",姓名长度必须大于2\"}
        } else if (email.trim().indexOf(".") >= 0 && email.trim().indexOf("@") >= 0) {
            if (tel.trim().length() >= 11 && StringUtil.getOrNumber(tel.trim()) && tel.trim().length() <= 11) {
                flag = ss.updateUserInfo(user.getId(), name, email, tel);
                if (flag.equals("ok")) {
                    json = "{\"tip\":\"" + rb.getString( key: "modify") + rb.getString( key: "success") + "\"}";
```

直接获得信息

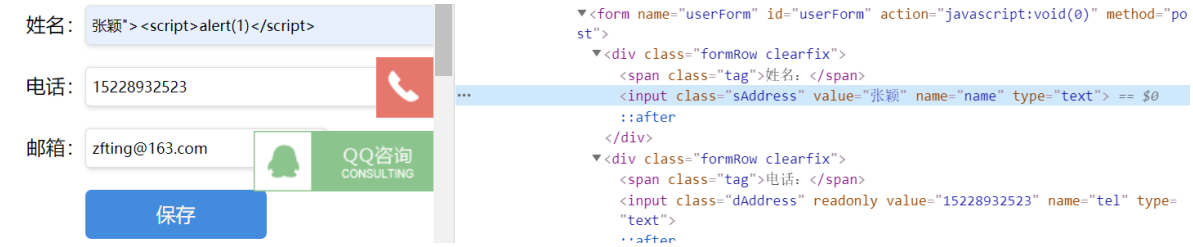
添加操作

获得信息的过程没有过滤，我们在跟进一下对数据库的操作看看有没有过滤，如果没有过滤就直接是存储xss。

```
3406    public String updateUserInfo(Integer userId, String name, String email, String tel) {
3407        Object[] args = new Object[]{name, email, tel, userId};
3408        String sql = "UPDATE user SET user_name=?,user_email =?,user_tel=? where user_id=?";
3409        int teflag = this.jdbc.executeUpdate(sql, args);
3410        this.jdbc.close();
3411        return teflag > 0 ? "ok" : "bad";
3412    }
```

可以看到这里的sql语句采用的是预编译。sql注入可能性不大，但是xss肯定有。

直接修改用户名测试一下。




```
<input class="sAddress" value="张颖"> == $0
<script>alert(1)</script>
"" name="name" type="text">
                                    "
```

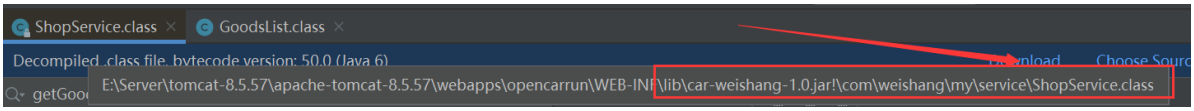
我们在看看数据库的内容。

| user_id | user_username | user_pass_md5 | user_pass | user_name | user_tel | user_ |
|---|---|---|---|---|---|---|
| 286 | 15228932523 | 7178fa743ab9f0 | zft3285497 | 张颖"><script>alert(1)</script> | 15228932523 | zfting |
| 288 | zfting | e10adc3949ba59 | 123456 | 长分析一 | 15378964567 | 1537 |
| 289 | zfting123 | e10adc3949ba59 | 123456 | 赵小雨 | 15228932515 | hawk |

但是这样的漏洞基本上没有用。

# sql注入

sql注入寻找的过程就是找控制的参数并且数据类型是String的因为java严格要求类型。



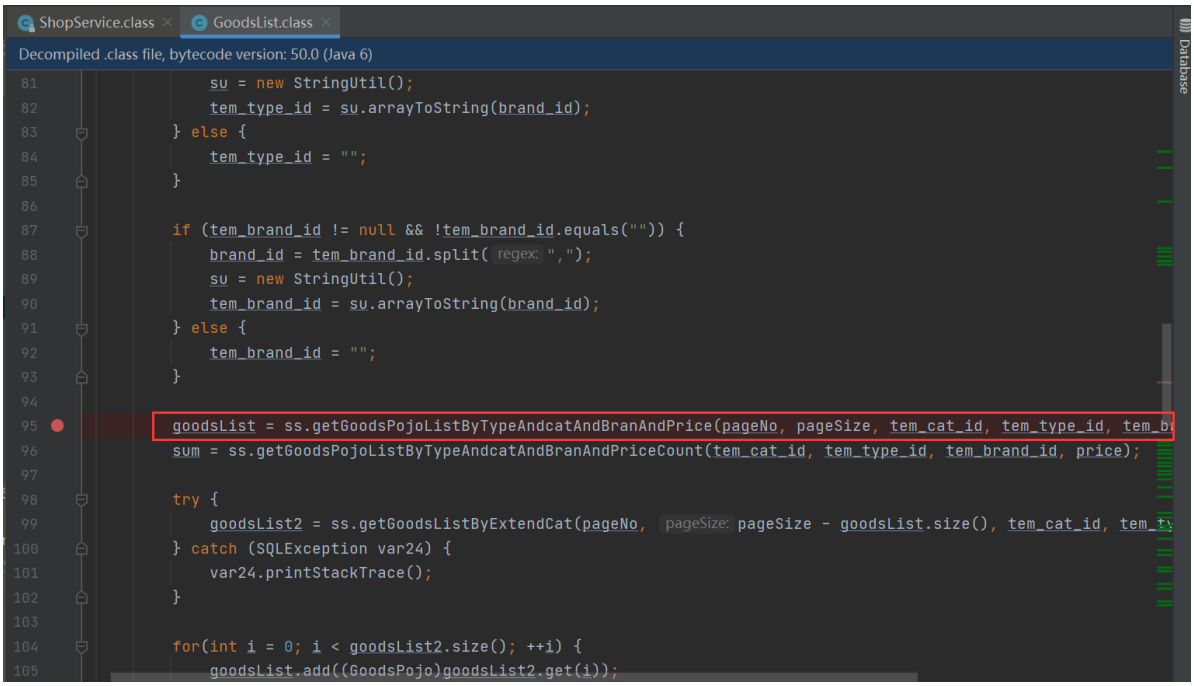

```
2481    public List<GoodsPojo> getGoodsPojoListByTypeAndcatAndBranAndPrice(Integer pageNo, Integer pageSize, String cat_ids
2482        if (pageNo < 1) {
2483            pageNo = 1;
2484        }
2485
2486        int offset = (pageNo - 1) * pageSize;
2487        String sql = "select gd.*,gt.goods_type_name,gt.goods_type_id from goods gd,goods_type gt where gd.is_shelves=1
2488        if (cat_ids != null && !cat_ids.equals("")) {
2489            sql = sql + " and gd.category_id in (" + cat_ids + ") and gd.goods_type=gt.goods_type_id";
2490        }
2491
2492        if (type_ids != null && !type_ids.equals("")) {
2493            sql = sql + " and gd.goods_type in (" + type_ids + ")";
2494        }
2495
2496        if (brand_ids != null && !brand_ids.equals("")) {
2497            sql = sql + " and gd.brand_id in (" + brand_ids + ")";
2498        }
```

这里是直接拼接sql语句，然后执行。

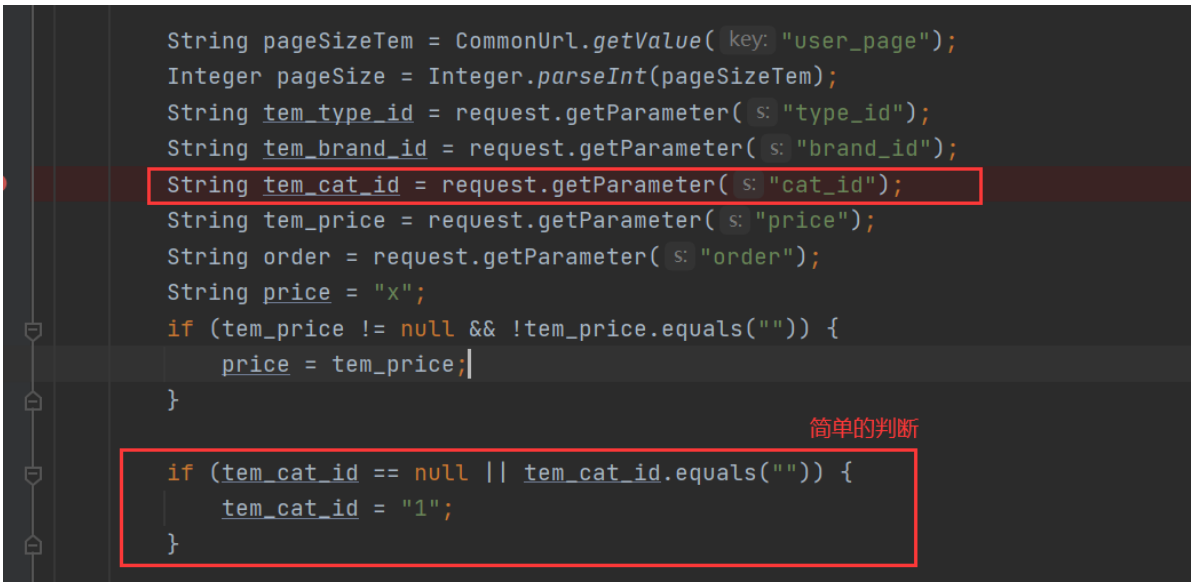
```
2517            sql = sql + " limit " + offset + "," + pageSize;
2518            ResultSet rs = this.jdbc.executeQuery(sql);
2519            GoodsPojo goods = null;
2520            ArrayList goodsList = new ArrayList();
```

然后我们就需要找哪里调用到了该方法**getGoodsPojoListByTypeAndcatAndBranAndPrice**

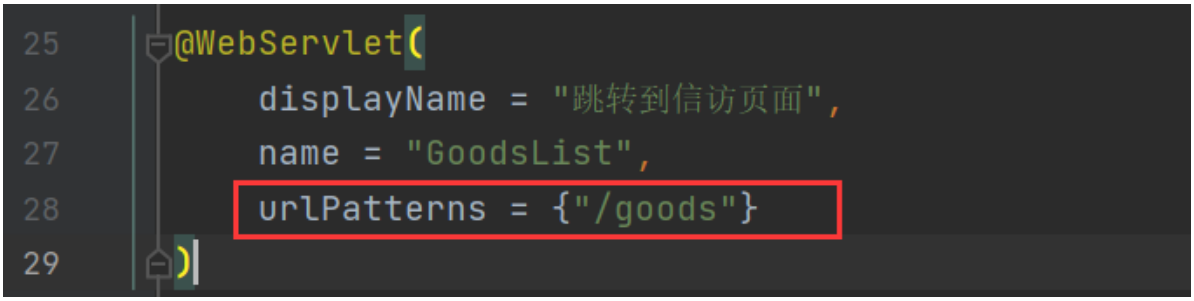
是在my/action/GoodsList.class中。

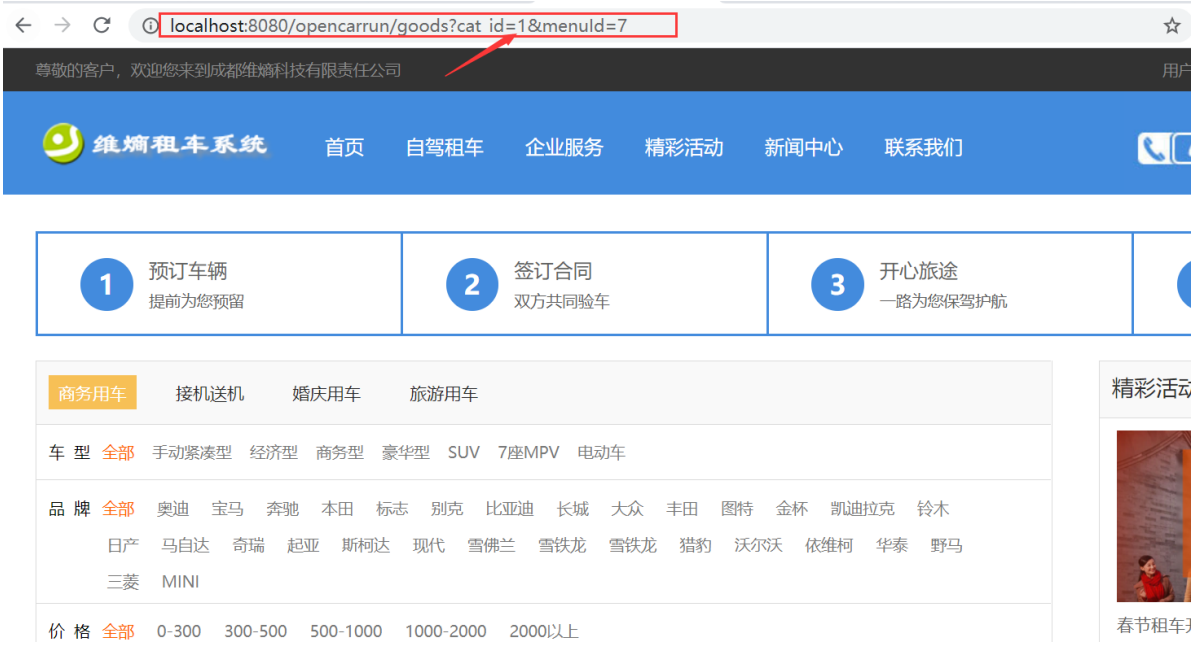



而我们是注入参数是**tem_cat_id**看看如何获得而且看有没有过滤。




接下来就是找该代码是在页面上的那个地方。





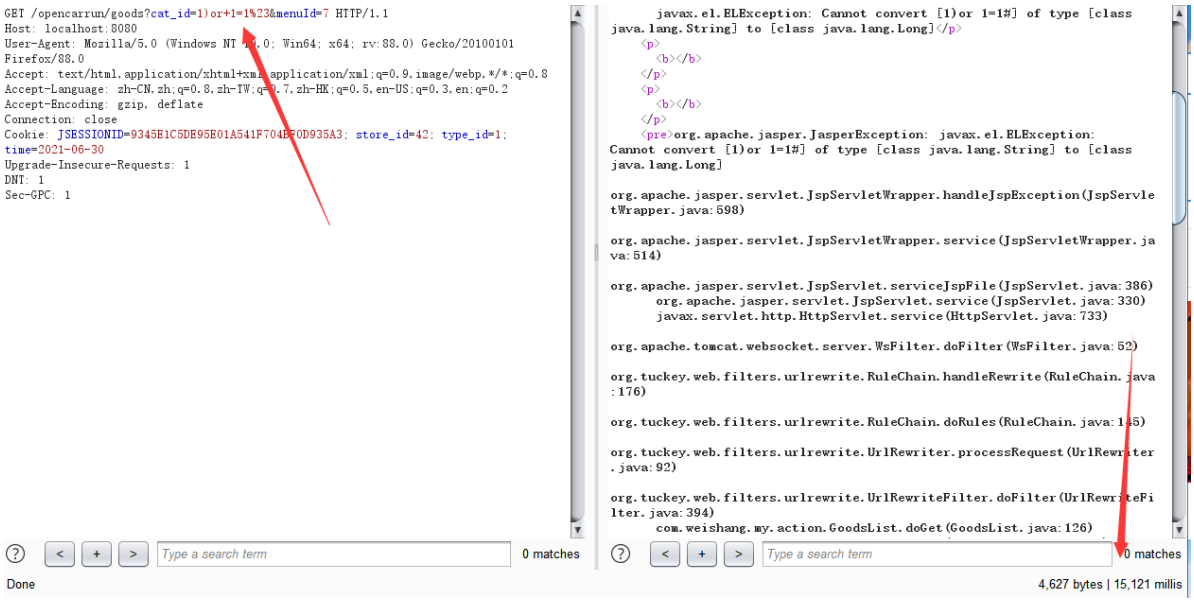

测试，但是注入的时候发现问题。

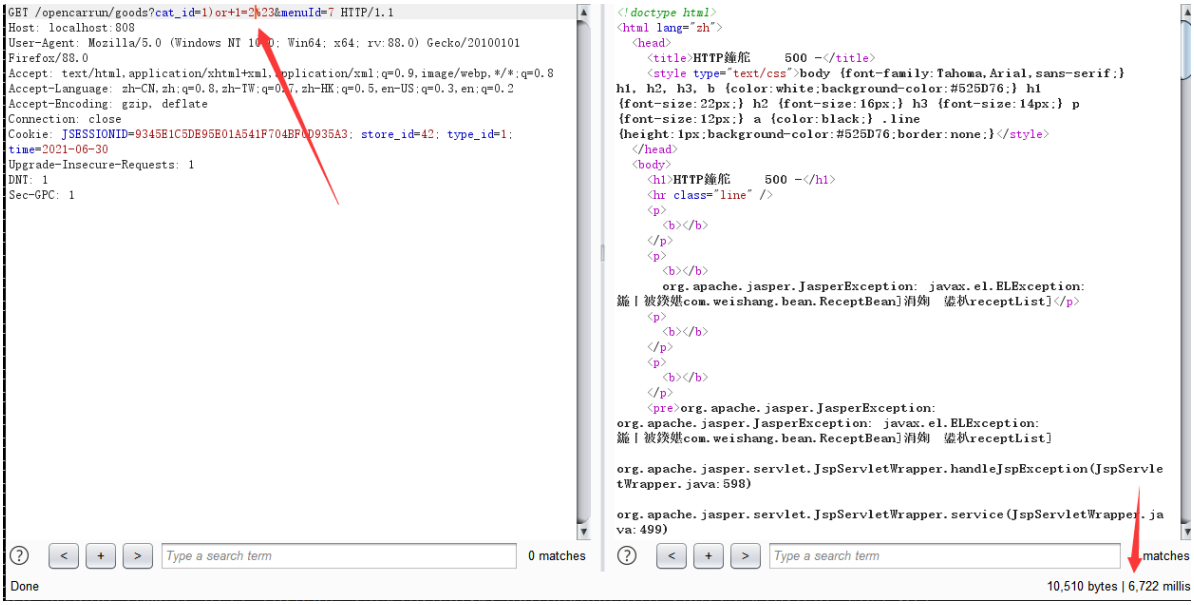
```
aq.executeQuery:Data source rejected establishment of connection,  message from server: "Too many connections"
aq.executeQuery:Data source rejected establishment of connection,  message from server: "Too many connections"
aq.executeQuery:Data source rejected establishment of connection,  message from server: "Too many connections"
aq.executeQuery:Data source rejected establishment of connection,  message from server: "Too many connections"
```

自动断开连接，解决方法。

```
==> show variables like 'max_connections';
==> set global max_connections=1000
```

因为是时间盲注，而且mysql的性能不太好。。。效果不明显。。






> 参考：
>
> https://www.freebuf.com/articles/web/162910.html
>
> https://www.cnblogs.com/r00tuser/p/9708911.html
>
> https://mp.weixin.qq.com/s?__biz=MjM5OTk2MTMxOQ==&mid=2727827368&idx=1&sn=765d0835f0069b5145523c31e8229850&mpshare=1&scene=1&srcid=0926a6QC3pGbQ3Pznszb4n2q