某次红蓝对抗之 Solr-RCE 实战绕过 - 先知社区

> 先知社区，先知安全技术社区

前言

在某次红蓝对抗过程中。

要结束的时候突然发现了扫描器爆出了

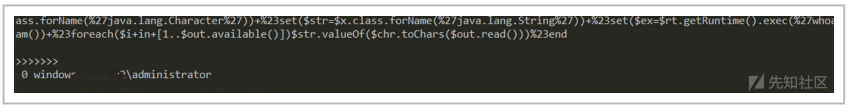
Solr-RCE-CVE-2019-0192 漏洞。

但是进行测试过程中, 发现存在各种各样的问题

(https://xzfile.aliyuncs.com/media/upload/picture/20220714102337-f7e457dc-031b-1.png)

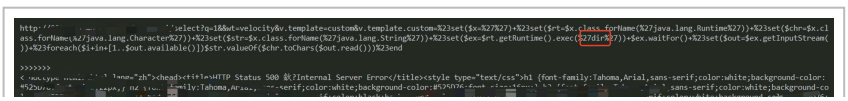
绕过 1

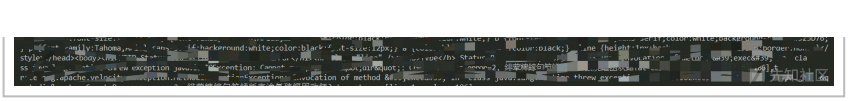
进行测试, 发现目标只可以执行单命令, 返回字段比较少的命令。



(https://xzfile.aliyuncs.com/media/upload/picture/20220714102455-2616347c-031c-1.png)

执行 dir, 无法执行。





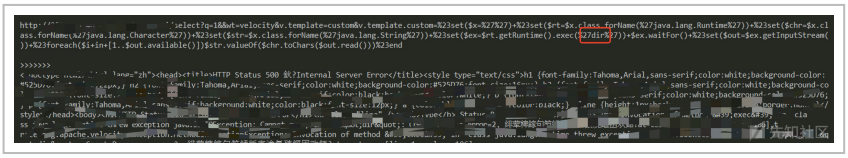
(https://xzfile.aliyuncs.com/media/upload/picture/20220714102533-3cdc40fc-031c-1.png)

不出网

想着直接执行 ps 直接上线就好了，各种尝试之后，后知后觉发现对方不出网

写 websgell

发现目标不出网的时候, 只有写 webshell 这一条路子可以走了。
但是目标只能执行个别命令还无法解决。



(https://xzfile.aliyuncs.com/media/upload/picture/20220714102647-692f88f8-031c-1.png)

dir 无法执行。陷入了沉思，加入单双引号也不行。
根据以前的内网经验是不是系统无法默认调用到 dir.exe。
那么 cmd /c dir 是不是可以。

(https://xzfile.aliyuncs.com/media/upload/picture/20220714102713-785d56e8-031c-1.png)
惊奇的发现, 可以完美的执行命令。
通过 dir, 找到目录。进行写马尝试。
但是发现目标路由规则写死了, 无法直接访问到. jsp 的文件。

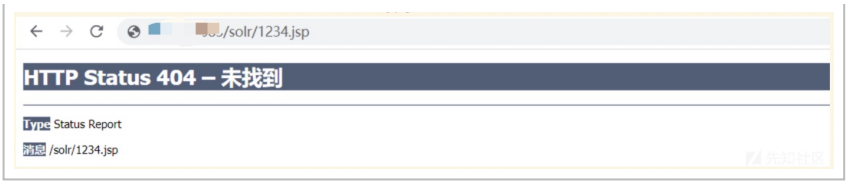
(https://xzfile.aliyuncs.com/media/upload/picture/20220714102730-82a0df8a-031c-1.png)
刚开始以为是根目录的问题。
发现不止根目录，常用的 css/js 和 img 下面的也不行。

(https://xzfile.aliyuncs.com/media/upload/picture/20220714102743-8aa76bd6-031c-1.png)
后续在 webshell 中看到，翻文件看到了有类似路由机制的验证

(https://xzfile.aliyuncs.com/media/upload/picture/20220714102759-941b17b2-031c-1.png)

言归正传

在执行 rce 的时候，找到了 solr 的目录。发现这里的. jsp 是没有这个验证的。



(https://xzfile.aliyuncs.com/media/upload/picture/20220714102819-9fed5398-031c-1.png)
利用命令执行进行找到该位置，进行写文件。
问题来了。echo 写入一直无法写入。。

(https://xzfile.aliyuncs.com/media/upload/picture/20220714102832-a7e641cc-031c-1.png)
问题解决

> 把这里的特殊字符进行特殊字符编码，即可成功写入。

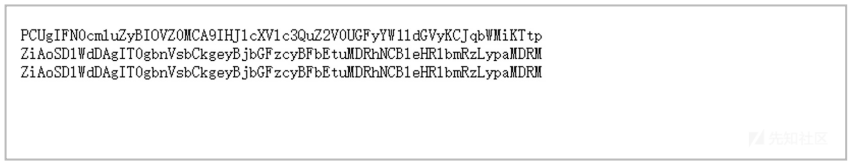
又遇到一个问题。jsp 的马子都有 % 号，这里不论怎么做 % 就是写不进去。
差点放弃。找不到不带 % 的马子。

柳暗花明

但是想到了上午利用过的 Certutil 可以进行编码解码
这样就没有特殊字符了。

(https://xzfile.aliyuncs.com/media/upload/picture/20220714102915-c17bbb44-031c-1.png)
完全没问题。

刚开始一点一点追加，发现下面的会写进去两行



(https://xzfile.aliyuncs.com/media/upload/picture/20220714102927-c87dbeb0-031c-1.png)

看了一下就最后有一个 + 号，没有特殊字符。
全部直接写进去。

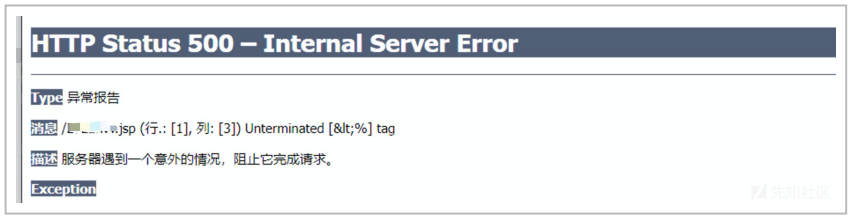
(https://xzfile.aliyuncs.com/media/upload/picture/20220714103206-275ca4a0-031d-1.png)

然后 decode 进行解码，完美。

(https://xzfile.aliyuncs.com/media/upload/picture/20220714103214-2be11c2c-031d-1.png)

(https://xzfile.aliyuncs.com/media/upload/picture/20220714103219-2eecfbb6-031d-1.png)

访问但是是 500. 不过确很开心，因为确实写上来。



(https://xzfile.aliyuncs.com/media/upload/picture/20220714103232-36ad5486-031d-1.png)

接下来解决为啥 500 就可以了。

type 123.jsp

查看一下。

(https://xzfile.aliyuncs.com/media/upload/picture/20220714103247-3f779b08-031d-1.png)

发现最后 decode 的时候，少了一个 >

本地测试，是没有这个问题的。可能是目标一次性字符长度的问题。

这里很简单了。

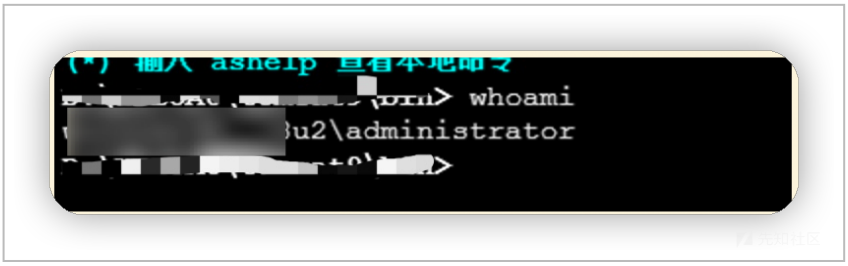
追加一下就可以了。

(https://xzfile.aliyuncs.com/media/upload/picture/20220714103302-48b01d6c-031d-1.png)

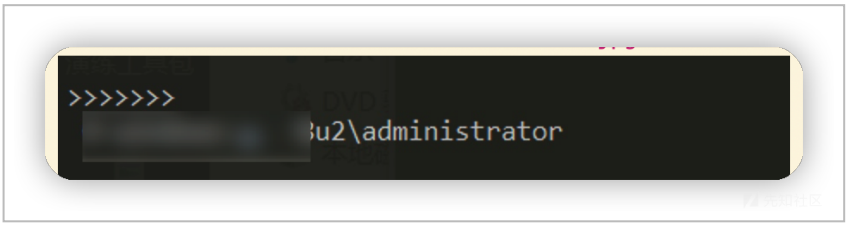
连接成功

(https://xzfile.aliyuncs.com/media/upload/picture/20220714103309-4cc15042-031d-1.png)

验证



(https://xzfile.aliyuncs.com/media/upload/picture/20220714103404-6da2e0d2-031d-1.png)



(https://xzfile.aliyuncs.com/media/upload/picture/20220714103423-78a8c8f2-031d-1.png)