Smarty <= 3.1.32 PHP代码执行漏洞分析—【CVE-2017-1000480】

chybeta / 2018-01-24 09:38:00 / 浏览数 6496 技术文章 技术文章 顶(2) 踩(0)

smarty简介

smarty是一个php模板引擎，其项目地址：https://github.com/smarty-php/smarty。

smarty据有模板编译功能。当访问一个模板文件时，smarty会根据模板文件在设置的编译目录中生成对应的php脚本（即编译文件），此后若再次访问该模板文件时，倘若

环境搭建

测试环境：linux

根据commit信息，我们检出 6768340，此时漏洞还未修复。

```
λ git clone https://github.com/smarty-php/smarty.git
λ cd smarty\
λ git checkout 6768340
λ cd ..
```

index.php:

```
<?php
include_once('./smarty/libs/Smarty.class.php');
define('SMARTY_COMPILE_DIR','/tmp/templates_c');
define('SMARTY_CACHE_DIR','/tmp/cache');

class test extends Smarty_Resource_Custom
{
   protected function fetch($name,&$source,&$mtime)
   {
       $template = "CVE-2017-1000480 smarty PHP code injection";
       $source = $template;
       $mtime = time();
   }
}

$smarty = new Smarty();
$my_security_policy = new Smarty_Security($smarty);
$my_security_policy->php_functions = null;
$my_security_policy->php_handling = Smarty::PHP_REMOVE;
$my_security_policy->modifiers = array();
$smarty->enableSecurity($my_security_policy);
$smarty->setCacheDir(SMARTY_CACHE_DIR);
$smarty->setCompileDir(SMARTY_COMPILE_DIR);
$smarty->registerResource('test',new test);
$smarty->display('test:'.$_GET['chybeta']);
?>
```

漏洞分析

参数通过`$smarty->display('test:'.$_GET['chybeta']);`传入，`display`定义在 smarty_internal_templatebase.php 中，它调用了 `_execute`。

`_execute`定义在`libs\sysplugins\smarty_internal_compile_assign.phpsmarty_internal_templatebase.php` 的 156
行左右，在该方法定义中，也即整个文件的174行左右：

```
# smarty_internal_templatebase.php
# line about 175

$template = $smarty->createTemplate($template, $cache_id, $compile_id, $parent ? $parent : $this, false);
```

会调用`createTemplate`方法，将我们的传入的参数创建成一个模板，

接着会调用render方法，进行模板渲染。

```
# smarty_internal_templatebase.php
# line about 174

$result = $template->render(false, $function);
```

render方法定义在libs\sysplugins\smarty_template_compiled.php中，第105行开始对前面生成的模板进行处理：

```
# smarty_template_compiled
# line about 104

if (!$this->processed) {
   $this->process($_template);
}
```

process方法定义在第131行。现在初次访问，也即文件的第138行会对模板文件进行编译，即如简介中所言开始生成编译文件：

```
if (!$this->exists || $smarty->force_compile ||
   ($smarty->compile_check && $source->getTimeStamp() > $this->getTimeStamp())
) {
   $this->compileTemplateSource($_smarty_tpl);
   $compileCheck = $smarty->compile_check;
   $smarty->compile_check = false;
   $this->loadCompiledTemplate($_smarty_tpl);
   $smarty->compile_check = $compileCheck;
}
```

compileTemplateSource方法定义在同文件的第169行，在第181行装载完编译器后（loadCompiler()），调用write方法进行写操作：

```
public function compileTemplateSource(Smarty_Internal_Template $_template)
{
   ...
   try {
       // call compiler
       $_template->loadCompiler();
       $this->write($_template, $_template->compiler->compileTemplate($_template));
   }
   ...
```

跟入compileTemplate方法，定义libs\sysplugins\smarty_internal_templatecompilerbase.php第334行：

```
public function compileTemplate(Smarty_Internal_Template $template, $nocache = null,
                                Smarty_Internal_TemplateCompilerBase $parent_compiler = null)
{
   // get code frame of compiled template
   $_compiled_code = $template->smarty->ext->_codeFrame->create($template,
                                                                $this->compileTemplateSource($template, $nocache,
                                                                                             $parent_compiler),
                                                                $this->postFilter($this->blockOrFunctionCode) .
                                                                join('', $this->mergedSubTemplatesCode), false,
                                                                $this);
   return $_compiled_code;
}
```

create是生成编译文件代码的方法，定义在libs\sysplugins\smarty_internal_runtime_codeframe.php第28行，为显示变量情况，这里我加了一句var_dump

在第44行，在生成output内容时有如下代码：

```
$output .= "/* Smarty version " . Smarty::SMARTY_VERSION . ", created on " . strftime("%Y-%m-%d %H:%M:%S") .
                  "\n  from \"" . $_template->source->filepath . "\" */\n\n";
```

将 $_template->source->filepath的内容直接拼接到了$output里。这段代码是为了生成编译文件中的注释，$output的头尾有注释符号/*和*/。

现在考虑如何利用，我们需要闭合前面的注释符号，即payload的最前面需要加上*/。同时还要把后面的*/给注释掉，可以在payload最后加上//。中间填上php代码即可。

在linux平台下即可利用成功。

漏洞修补

查看commit记录：https://github.com/smarty-php/smarty/commit/614ad1f8b9b00086efc123e49b7bb8efbfa81b61

添加了过滤，将可能闭合的*/变为* /：

在另外几处文件中也进行了过滤，要求只能出现字母和数字:

```
substr(preg_replace('/[^A-Za-z0-9.]/','',$source->name),0,25);
```

题外话

直接看生成的编译文件，会发现有两个输出点，第二个输出点在单引号内，但这个无法逃逸。在libs\sysplugins\smarty_internal_runtime_codeframe.php的第

```
$dec = "\$_smarty_tpl->_decodeProperties(\$_smarty_tpl, " . var_export($properties, true) . ',' .
      ($cache ? 'true' : 'false') . ")";
$output .= "if ({$dec}) {\n";
```

而如漏洞修补一节中所言，添加过滤后，引号会被直接去除。