PHP: Better Webhosting Security - Disable exec(), exec_shell(), system(), popen(), eval() ... shell fork functions

Author : admin

If you work as System Administrator of WebHosting company, you definitely know how often it is that some *automated cracker scripts (working as worms) intrude through buggy old crappy custom coded sites or unupdated obsolete Joomla / Wordpress etc. installs*. and run themselves trying to harvest for other vulnerable hosts. By default PHP enables running commands via shell with PHP functions like exec();, shell_exec(); , system();. and those script kiddie scripts use mainly this functions to *spawn shell via vulnerable PHP app*. Then scripts use whether php curl support is installed (i.e. **php5-curl**) to download and replicate itself to next vulnerable hop.

With that said it is a must after installing new Linux based server for hosting to disable this functions, to save yourself from future hassles ...
Earlier, I blogged how to disable PHP system system(); and exec(); functions to raise Apache security using suhosin however this method requires php suhosin being used.

Yesterday, I had to **configure new web hosting server** with *Debian 7*, so I tried installing suhosin to use it to protect PHP from having enabled dangerous *system();, eval(); exec();* .
I remember disabling *system();* using suhosin php extension was working fine on older Debian releases, however in *Debian 6.0*, **php5-suhosin** package was causing *severe Apache crashes* and probably that's why in latest *Debian Wheezy 7.0*, php suhosin extension is no longer available. Therefore using suhosin method to disable *system();, exec(); and other fork* functions is no longer possible in Debian.

Since, **suhosin** is no longer there, I decided to use conventional PHP method via *php.ini.*

Here is how to do it

Edit:

/etc/php5/apache2/php.ini

```
debian:~# vim /etc/php5/apache2/php.ini
```

And near end of file placed:

**disable_functions =exec,passthru,shell_exec,system,proc_open,
popen,curl_exec, curl_multi_exec,parse_ini_file,show_source**

**allow_url_fopen Off
allow_url_include Off**

It is good to explain few of above functions - **shell_exec, proc_open, popen, allow_url_fopen,show_source and allow_url_include**.

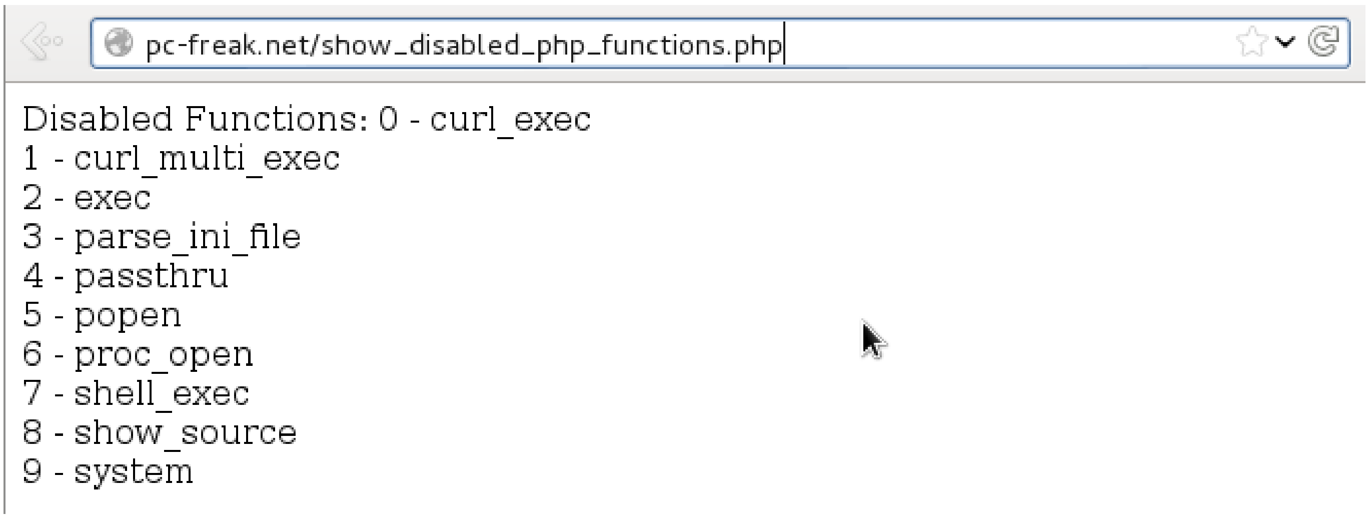
Disabling *shell_exec* - disables from PHP scripts executing commands with bash slash ` `, i.e. **`ls`**.
proc_open and **popen** allows reading files from file system.

show_source - makes possible also reading other PHP source files or can be used to display content of other files from fs.

To read newly placed config vars in *php.ini* usual apache restart is necessary*:*

```
debian:~# /etc/init.d/apache2 restart
[....] Restarting web server: apache2
. ok
```

Further on tÐ¾ test whether system();, exec();, passthru(); ... etc. are disabled. Make new PHP file with content:



pc-freak.net/show_disabled_php_functions.php

```
Disabled Functions: 0 - curl_exec
1 - curl_multi_exec
2 - exec
3 - parse_ini_file
4 - passthru
5 - popen
6 - proc_open
7 - shell_exec
8 - show_source
9 - system
```

Copy of above source code [show_disabled_php_functions.php is here for download](show_disabled_php_functions.php)

. To test your Apache PHP configuration disabled functions download it with **wget** or **curl** and rename it to .php:

```
 # cd /var/www # wget -q http://www.pc-freak.net/files/show_disabled_php_functions.php.txt
mv show_disabled_php_functions.php.txt show_disabled_php_functions.php
```

After disabling functions on those *newly setup Debian hosting Apache webserver*, I remembered, same functions were still active on another *CentOS Linux server*.

To disable it there as well, had to edit:

/etc/php.ini

```
 [root@centos:~]# vim /etc/php.ini
```

And again place after last file line;

```
 disable_functions =exec,passthru,shell_exec,system,proc_open,popen,
curl_exec, curl_multi_exec,parse_ini_file,show_source


allow_url_fopen Off
allow_url_include Off
```

Finally on CentOS host, had to restart Apache:

```
[root@centos:~]# /etc/init.d/httpd restart
```

For Security paranoids, there are plenty of other PHP functions to disable including, basic functions like **ln, mv, mkdir, cp, cat** etc.

Below is list of all functions to disable - only disable this whether you you're a PHP security freak and you're 100% some server hosted website will not use it:

```
disable_functions = "ln, cat, popen, pclose, posix_getpwuid, posix_getgrgid, posix_kill,
parse_perms, system, dl, passthru, exec, shell_exec, popen, proc_close, proc_get_status, proc_nice,
proc_open, escapeshellcmd, escapeshellarg, show_source, posix_mkfifo, mysql_list_dbs,
get_current_user, getmyuid, pconnect, link, symlink, pcntl_exec, ini_alter, pfsockopen, leak,
apache_child_terminate, posix_kill, posix_setpgid, posix_setsid, posix_setuid, proc_terminate,
syslog, fpassthru, stream_select, socket_select, socket_create, socket_create_listen,
socket_create_pair, socket_listen, socket_accept, socket_bind, socket_strerror, pcntl_fork,
pcntl_signal, pcntl_waitpid, pcntl_wexitstatus, pcntl_wifexited, pcntl_wifsignaled,
pcntl_wifstopped, pcntl_wstopsig, pcntl_wtermsig, openlog, apache_get_modules,
```

apache_get_version, apache_getenv, apache_note, apache_setenv, virtual, chmod, file_upload, delete, deleted, edit, fwrite, cmd, rename, unlink, mkdir, mv, touch, cp, cd, pico"