

Forum:[Hack Forums](#)

Thread: [executing php code on log file with file inclusion ?](#)

bdkalkun - September 1, 2018, 4:51 am

hi guys

so there's a host which i found only one vuln entry which is LFI

everything is accesible , proc/envron, etc/passwd ,usr/local/apache/logs/access_log but the entry point is on download page

for example host:80/download.php?file=

so my entry is similar like this host:80/download.php?file=/usr/local/apache/logs/access_log , but instead executing the php code, its downloading those log file

i did see the injected php code in the log file (php code is present)

any suggestion so i can executed those code?

Punisher - September 2, 2018, 10:47 am

The server doesn't know there's PHP in there - it's rendering it as a TXT document

You have a few options -

1. Figure out how to inject into the .htaccess log and allow txt or log documents to execute PHP by adding this line to the .htaccess:

AddHandler application/x-httpd-php .php .txt

Replace .txt with whatever file you managed to inject into (ie .html, .log, etc)

2. Try a null url inclusion:

host:80/download.php?file=/usr/local/apache/logs/access_log%00

3. Try to find the configuration file that contains the database username+password and download that, then simply log in and pwn

M L T - September 2, 2018, 9:26 pm

Are you certain this is local file inclusion and not just local file disclosure? the first (via include();) will treat the loaded page as an executable, whereas the latter will just display the page contents as plain text. From your description this most certainly sounds like a case of Local file DISCLOSURE rather than inclusion.

If this is an LFD (rather than an LFI) then really your only option would be to map out the local filesystem and see if you could pull some credentials from something - LDAP, MySQL, anything? Maybe you could see if /etc/shadow is readable or whether theres symlinks between passwd and shadowfiles (I know, dumb, but seen it many-a-time... trust me) - anyway, crack those shadow hashes if accessible (although goodluck if they're \$6\$ lol, hashcat might do it before the universe implodes if you're lucky. That being said, if they're \$1\$ hashes or something, then you should easily be able to crack them).

Ideally, scan the box for open ports/services, then test default locations for config files for those... for example if port 3306 is open and you can read my.cnf then you can just authenticate via MySQL. If there's readable \$1\$ shadow hashes you can just crack the root hash and SSH into the box -- remember just because you can't crack the root hash doesn't mean you can't crack other hashes. if you fail at cracking root hash then try cracking a user hash and performing priv esc from there, either horizontally or through means of a kernel exploit (speaking of which, what's the kernel version? you should be able to find it in /proc/issue?) -- If it's running a custom CMS, then with some enumeration and some luck you'll be able to grab the login credentials for the CMS and just find the admin panel from there and login (then hope for priv esc to full RCE via a vuln in the admin panel, as they tend to be less secure).

From your description, I am 99.9% sure this is an LFD rather than an LFI, meaning the files are treated as plaintext therefore any code you poison a logfile with will not be executed. Your only chance here is to gain access through means of fingerprinting and generic information disclosure (map out the OS, any CMS running, etc, then google to find the default paths for config files for these, or attempt to crack shadow hashes and see if you can escalate privs from there or crack root hash directly).

Also, to expand on the other posters' suggestion that you use a nullbyte (in the case this is actually an LFI and it's just not treating it as PHP due to removal of extension... my bet is that it's an LFD though and isn't a file inclusion whatsoever, just file disclosure), but just in case I'm wrong and it is LFI then it should be noted that nullbytes aren't the only method of preventing it from removing the PHP extension - you can attempt other control chars such as CR/LF or tabbing (%0d%0a, %09, etc) - you could also add a ? or & to the end of your request in order to trick the server into thinking it is about to receive another GET parameter.