Hiding data inside the padding area in files and packets

Fady Mohammed Osman
Pen. Tester
fady.mohamed.osman@gmail.com
September 2010

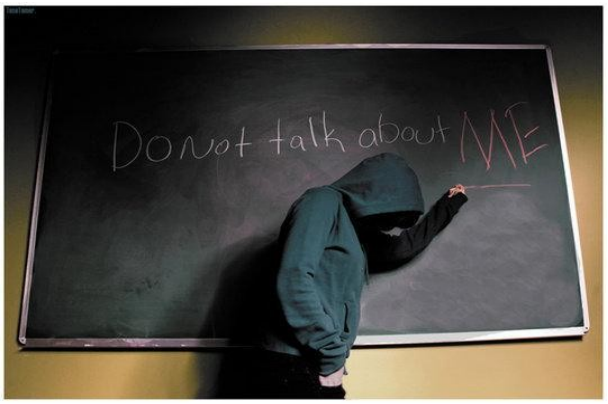
Hiding data inside the padding area of files

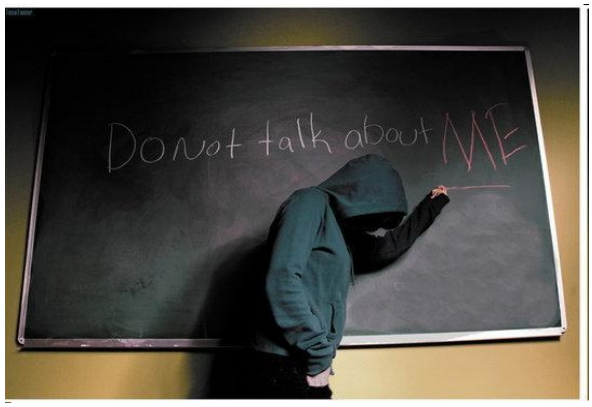
Today we will talk about steganography which is the science of hiding information inside other data instead of just encrypting it you can think of it as the cousin of cryptography and you can mix steganography with cryptography to have something stronger than both.

In most cases this can be done by changing small number of bytes in a file to contain your data. This can be an image file or sound file.

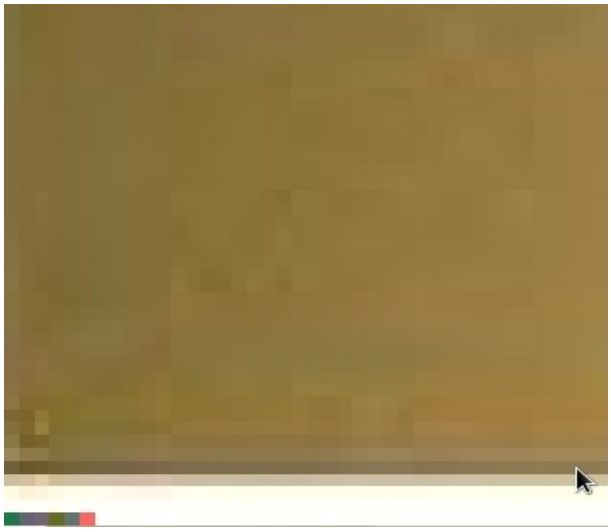
To make this clear lets see an example.

One of the following images contains the message “My secret message” can you tell which one of them contains the message???

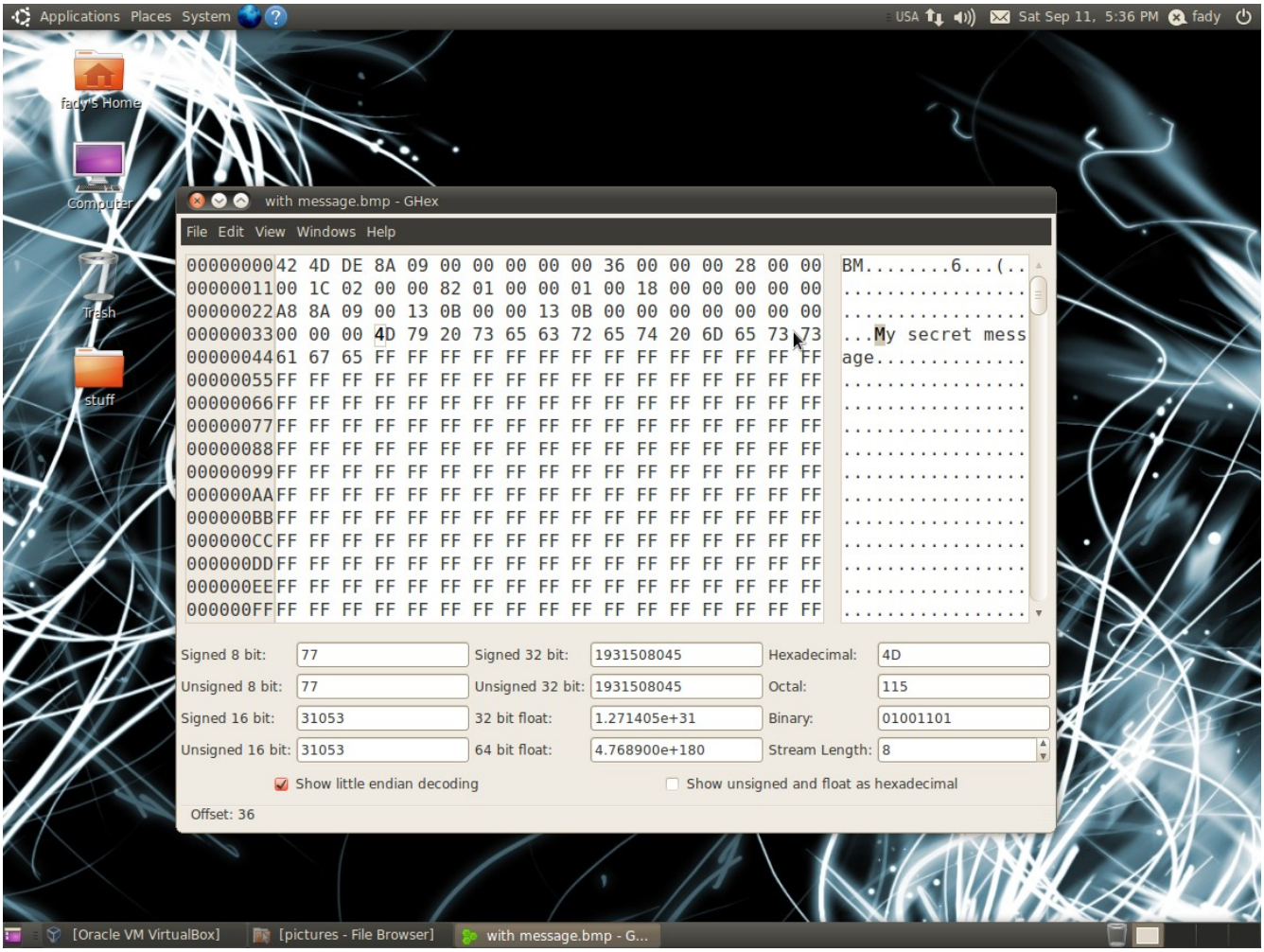




Actually yes, by zooming in the second image you will notice that the image have some colored pixels at the bottom left corner those pixels are the pixels that contains our message.



The following picture shows the image opened with a hex editor (you may use hex work shop in windows or ghex or bless hex editor in linux):



Now you can see our message.

Cool right. But what if some one else zooms in the image and he notices these colored pixels and by further examination he determined the message we hide??

One way to solve this is by encrypting the message but still it will be obvious that there's something strange in the image and by some cryptanalysis the cipher can be broken .and i have seen some applications using this technique.

Any way i came up with new idea i don't know if anybody uses this before or not.

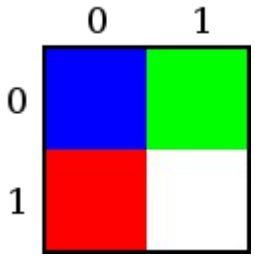
What we gonna do is that we will hide the information inside what is called the padding area so first lets talk about what is the padding area (you can skip this part if you are already familiar with the padding area).

What is the padding area??

It's some bytes added to a file or a network packet for 4 byte alignment. Those bytes are added because computer can handles data which are multiples of four bytes faster since the registers and the buses are 32 bits(assuming a 32 bits machine), also one of the obvious examples is your graphics card which needs the frames sent to it to be aligned for four bytes.

Let's examine a simple bmp file and see the padding area (this example was taken from bmp file specifications in wikipedia):

Here's the bmp:

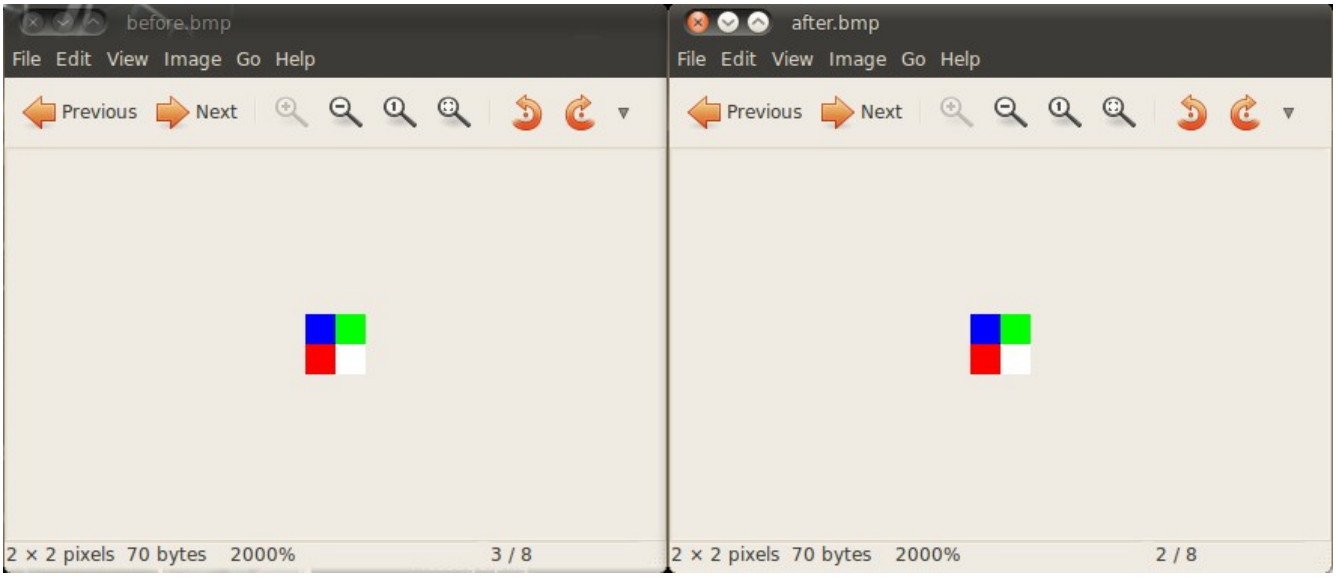


and this table shows how the image stored in the file:

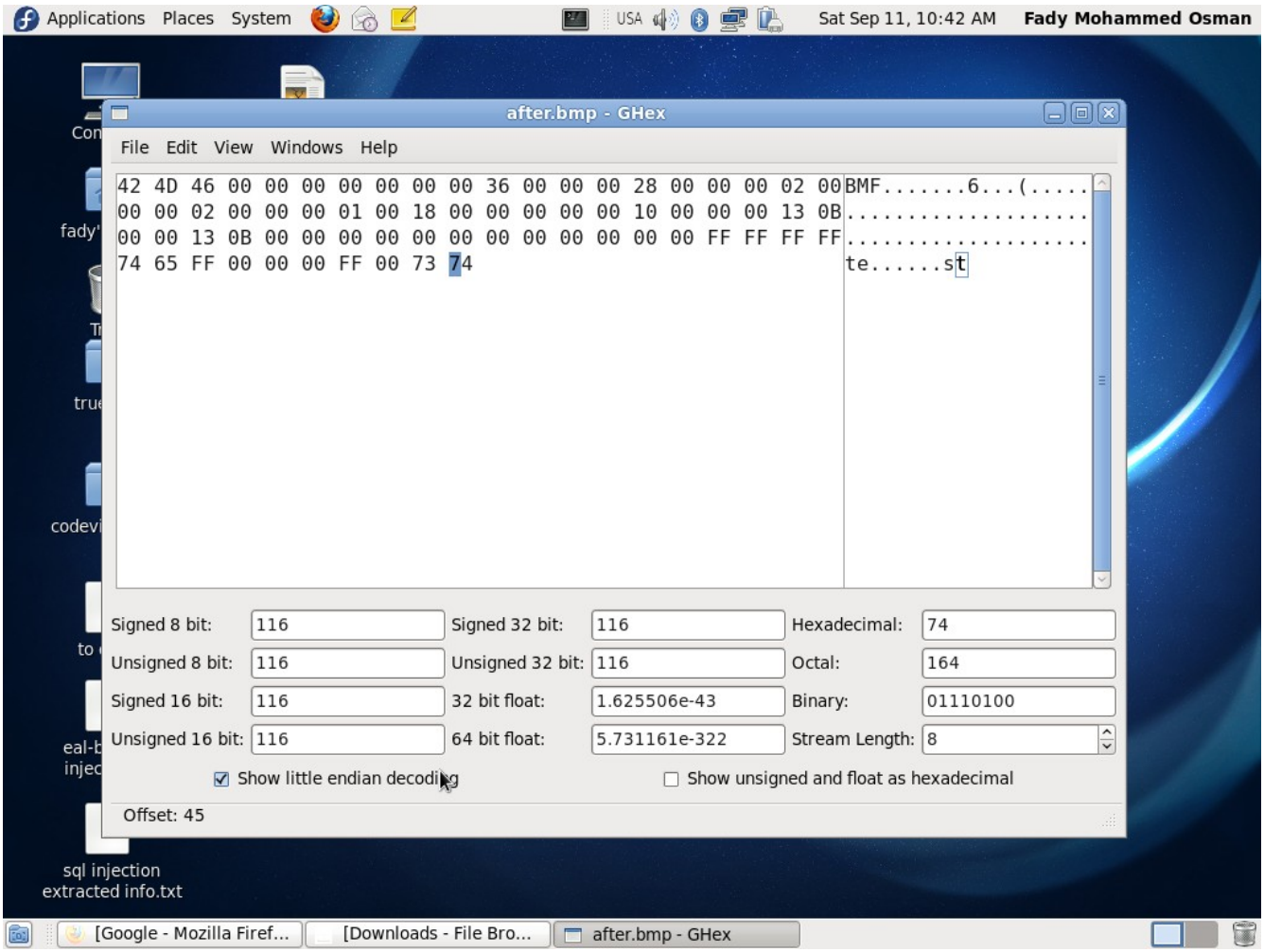
36h	3	00 00 FF	0 0 255	Red, Pixel (0,1)
39h	3	FF FF FF	255 255 255	White, Pixel (1,1)
3Ch	2	00 00	0 0	Padding for 4 byte alignment (Could be a value other than zero)
3Eh	3	FF 00 00	255 0 0	Blue, Pixel (0,0)
41h	3	00 FF 00	0 255 0	Green, Pixel (1,0)
44h	2	00 00	0 0	Padding for 4 byte alignment (Could be a value other than zero)

as you can see in an image of 4 pixels we have wasted four bytes and these bytes will not be rendered by any graphics application and no application cares about their value so we can hide our message on them.

Lets see an example in this 4 pixel image and store the word test in those four bytes and see if the colors changed. One of the following two images contains the message and the other one doesn't contain any thing:



as you can see the colors haven't changed and in the following image you can see the second image containing the word “test” opened in ghex:



How to calculate the available padding space??

you can calculate the available padding space using the following formula(assuming a 32-bit machine):

(number of pixels in a row * number of bytes for each pixel)%4

Where can i use this??

It can be used for hiding information but also it can be used for uploading shell codes in websites that allows image uploading and this can be used by worms instead of uploading the shell code to a server.

If i used it to store a shell code will the antivirus be able to detect that??

I think that antivirus can't detect that since the signature will be changed by changing the image or it's dimensions.