

STEGANOGRAPHY

How to Hide Secret Data Inside an Image or Audio File in Seconds

BY BLACK SLASH © 11/11/2017 7:48 PM 🔄 09/07/2018 5:55 AM

Steganography is the art of hiding information in plain sight, and in this tutorial, I'll show you how to use [Steghide](#) — a very simple command line tool to do just that. In addition, I'll go over a bit of conceptual background to help you understand what's going on behind the scenes. This is a tool that's simple, configurable, and only takes a few seconds to hide information in many file types.

What Is Steganography?

Unlike encryption, where it's obvious that a message is being hidden, [steganography hides data in plain view](#), inside a file such as a picture. As far as images are concerned, to anyone who isn't aware that it contains hidden data, it looks like just a normal, innocent picture.

Steganography is useful in situations where sending encrypted messages might raise suspicion, such as in countries where free speech is suppressed. It's also frequently used as a digital

watermark to find when images or audio files are stolen. And on a less practical note — it's just cool.

- **More Info:** [A Beginner's Guide to Steganography](#)

Conceal Secret Messages or Data Through Steganography with Steghide [How-To]

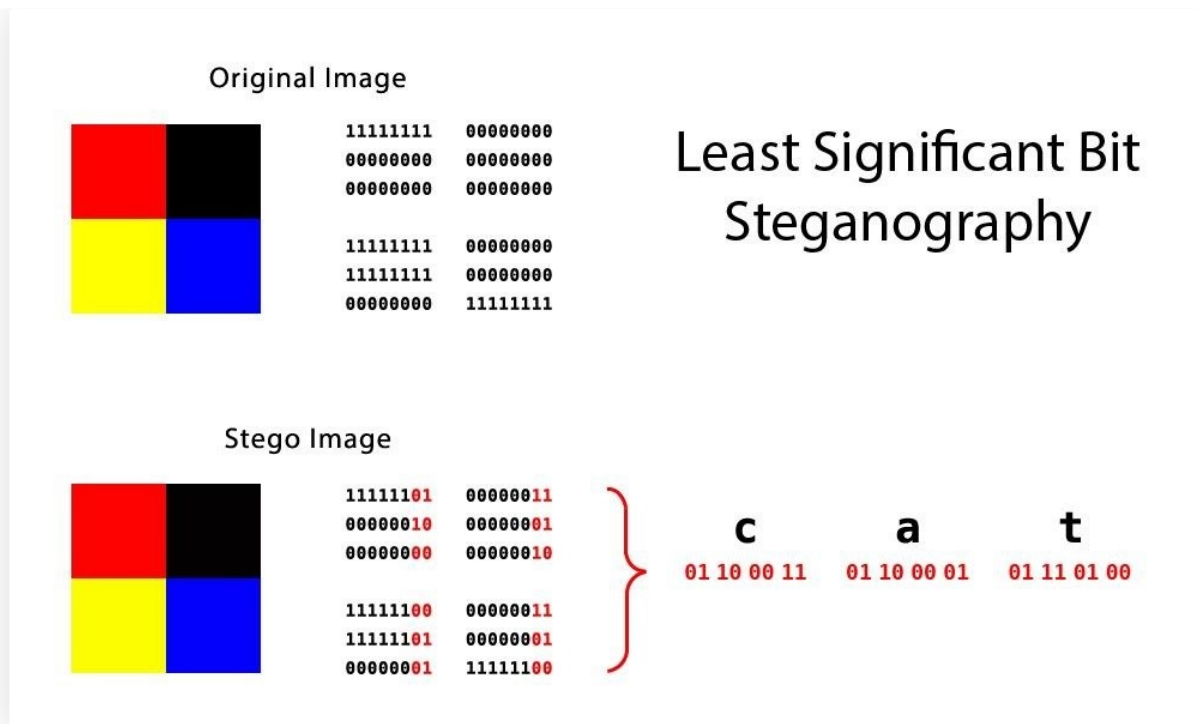


How Is Steganography Implemented?

There are several different techniques for concealing data inside of normal files. One of the most widely used and perhaps simplest to understand is the least significant bit technique, known commonly as LSB.

This technique changes the last few bits in a byte to encode a message, which is especially useful in something like an image, where the red, green, and blue values of each pixel are represented by eight bits (one byte) ranging from 0 to 255 in decimal or 00000000 to 11111111 in binary.

Changing the last two bits in a completely red pixel from 11111111 to 11111101 only changes the red value from 255 to 253, which to the naked eye creates a nearly imperceptible change in color but still allows us to encode data inside of the picture.



This diagram shows two 4-pixel images in both color and binary values. Each block of binary represents the value of the corresponding pixel.

The least significant bit technique works well for media files, where slightly changing byte values creates only slight imperceptible changes, but not so well for things like ASCII text, where a single bit out of place will completely change the character. That's not to mention the fact that data hidden using LSB steganography is also easy to detect if someone is looking for it.

For this reason, there are a plethora of other steganography techniques out there, each with their own benefits and drawbacks. Another far less detectable one is called the discrete cosine transform coefficient technique (I know, it's a mouthful), which slightly changes the weights (coefficients) of the cosine waves that are used to reconstruct a JPEG image.

Using Steganography Effectively

Keeping in mind that certain digital steganography techniques are better than others, generally, it's best to avoid the LSB technique and go for something a bit more sophisticated. In fact, designing your own steganography algorithm isn't terribly difficult if you already have good coding and math foundations. But to get a feel for how steganography works, LSB, which Steghide uses, will do just fine here.

Two other things to consider are encryption and compression. Encrypting data before embedding it adds an extra layer of security while compressing your data will obviously allow you to fit more into your cover file. Both encryption and compressions schemes can be included as optional parameters in Steghide, and we'll cover these below.

Embed Hidden Data into a File

Using Steghide is very easy. To install it from the terminal in Linux, just use [apt](#).

“ | `apt-get install steghide`

Once it's installed, in order to embed data in a file, type the command below.

“ | `steghide embed -ef secretFile -cf coverFile -sf outputFile -z compressionLevel -e scheme`

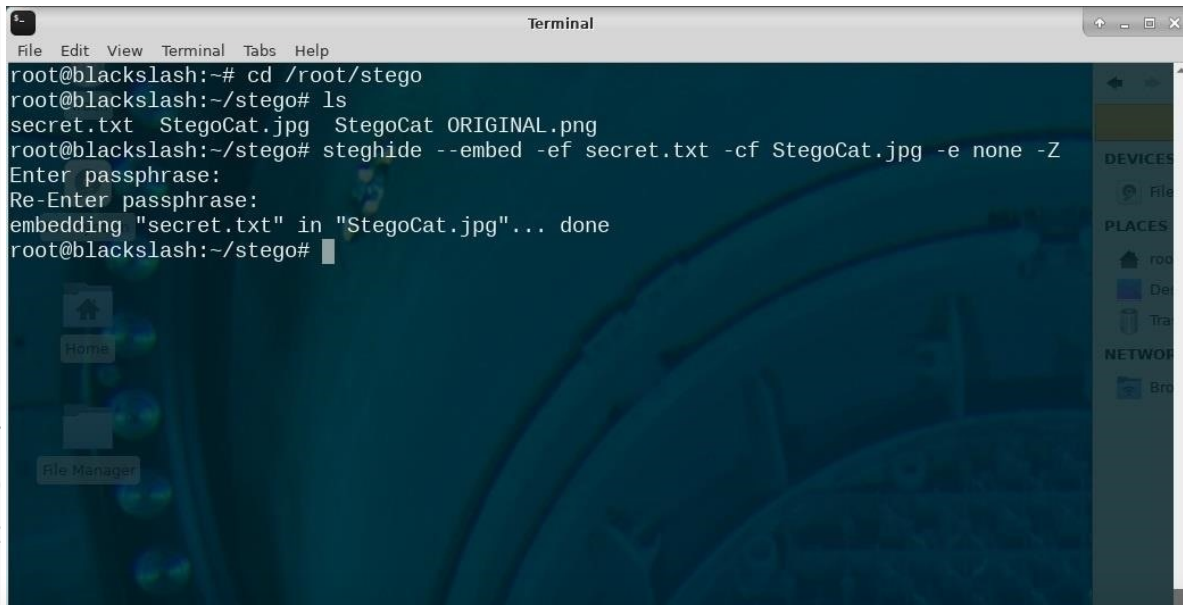
The arguments are broken down as follows:

- **-ef** specifies the path of the file that you want to hide. You can embed any kind of file inside of the cover file, including Python scripts or shell files.
- **-cf** is the file that the data is embedded into. This is restricted to BMP, JPEG, WAV, and AU files.
- **-sf** is an optional argument that specifies the output file. If this is omitted, the original cover file will be overwritten by your new steganographic file.
- **-z** specifies the compression level, between 1 and 9. If you prefer not to compress your file, use the argument **-Z** instead.
- **-e** specifies the type of encryption. Steghide supports a multitude of encryption schemes, and if this argument is omitted by default, Steghide will use 128-bit AES encryption. If you prefer not use encryption, simply type **-e none**

In my example, I'm hiding secret text inside an image of a cat. I'm not overwriting the original image or compressing it, nor do I care about encryption right now.

“ | `steghide embed -ef secret.txt -cf StegoCat.jpg -e none -Z`

- **Don't Miss:** [Hide Data in Audio Files \(Just Like Elliot in Mr. Robot\)](#)



```
root@blackslash:~# cd /root/stego
root@blackslash:~/stego# ls
secret.txt  StegoCat.jpg  StegoCat  ORIGINAL.png
root@blackslash:~/stego# steghide --embed -ef secret.txt -cf StegoCat.jpg -e none -Z
Enter passphrase:
Re-Enter passphrase:
embedding "secret.txt" in "StegoCat.jpg"... done
root@blackslash:~/stego#
```

word that
enter it to
inside an



Here is a side-by-side comparison of the original image and the steganographic image. Can you detect any difference?

Image by Hisashi/Flickr

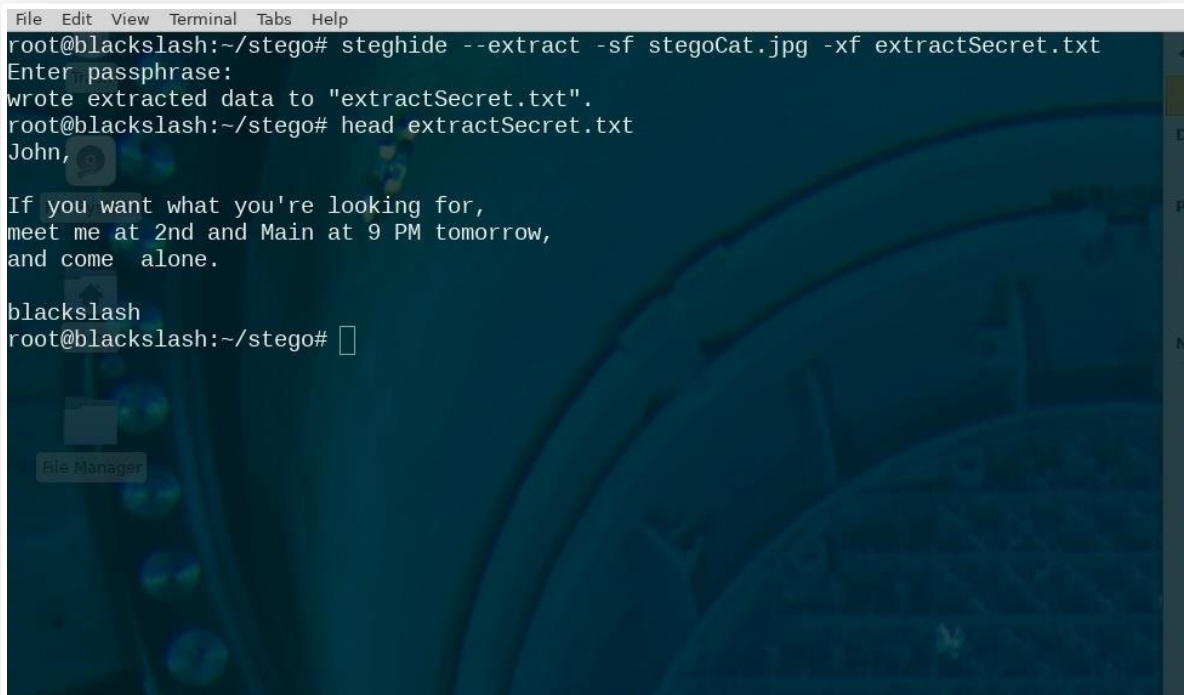
Step 2

Extract Hidden Data from the File

Extracting hidden data from a steganographic image is even easier. The command uses the syntax below.

“ \$ steghide extract -sf stegoFile -xf outputFile

Once you run this command, you'll be prompted to enter the same password you created above in order to create the extracted file. It's that simple!

A screenshot of a terminal window with a dark blue background and a faint image of a cat. The terminal shows the following commands and output:

```
root@blackslash:~/stego# steghide --extract -sf stegoCat.jpg -xf extractSecret.txt
Enter passphrase:
wrote extracted data to "extractSecret.txt".
root@blackslash:~/stego# head extractSecret.txt
John,

If you want what you're looking for,
meet me at 2nd and Main at 9 PM tomorrow,
and come alone.

blackslash
root@blackslash:~/stego#
```

I've extracted the stego data from the image to a file and displayed its contents in the terminal.

Hiding Data in Images Is Just Too Easy

The advantage of steganography is that you can hide data in plain sight, but you can really blow it if you don't follow some common sense rules. First, the small differences steganography introduces are hard to detect — unless you have the original.

Using an image you found off the internet without modifying it significantly makes it easy to detect that an image has hidden information. To check this, try a [reverse Google Image search](#) to make sure the original isn't floating around somewhere.

I hope you enjoyed this tutorial and realized just how easy it is to use steganography. It literally only takes a moment to hide secret messages inside media files, and whether you're doing it for copyright protection or just to be cool, steganography has a multitude of applications.

Thanks for reading, and if you have any questions, feel free to ask me in the comments below or on Twitter [@blackslash6](#).

Don't Miss: [How to Hack Forum Accounts with Password-Stealing Pictures](#)

- Follow Null Byte on [Twitter](#), [Flipboard](#), and [https://\www.youtube.com/channel/UCgTNupxATBfWmfehV21ym-g?sub_confirmation=1 YouTube
- Sign up for [Null Byte's weekly newsletter](#)
- Follow WonderHowTo on [Facebook](#), [Twitter](#), [Pinterest](#), and [Flipboard](#)

Cover photo by Black Slash/Null Byte (and Hisashi/Flickr); Screenshots by Black Slash/Null Byte

WonderHowTo.com About Us Privacy Policy Terms of Use