

ISM LAB-11

Name: Ayush Srivastava

Regno:19BCE1562

STEGANOGRAPHY

Steghide is a steganography tool that allows you to cover confidential records inside a picture or sound record with a passphrase. Bolsters BMP and JPEG picture group, AU and WAV sound group. This device has its advantages and disadvantages. One upside is that it is much better at covering and can extend a lot without any type of document. It does this by using a propelled calculation to shroud it inside a picture (or sound) record without changing the form (or sound) of the document. This is additionally without using Steghide (or if there is not the same scientific method as Steghide) then it is difficult to remove the hidden documents from the picture.

Steghide Installation

```
(root@kali)-[~]
# sudo apt-get install steghide -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  atril-common bubblewrap fonts-mathjax fonts-roboto-slab gir1.2-gtksource-3.0 gir1.2-javascriptcoregtk-4.0 gir1.2-soup-2.4 gnome-desktop3-data gobject-introspection
  libatrildocument3 libdjvulibre-text libdjvulibre21 libgnome-desktop-3-19 libgs9 libgs9-common libgxps2 libharfbuzz-icu0 libijs-0.35 libjavascriptcoregtk-4.0-18
  libjbig2dec0 libjs-mathjax libkpathsea6 libmanette-0.2-0 libpaper-utils libpaper1 libpipewire-0.3-0 libpipewire-0.3-common libpipewire-0.3-modules
  libsoup-gnome2.4-1 libspa-0.2-modules libspectrel libsyntax2 libwpe-1.0-1 libwpebackend-fdo-1.0-1 libxkbregistry0 pipewire pipewire-bin pipewire-media-session
  pwgen python-mpltoolkits.basemap-data python3-advancedhttpserver python3-boltions python3-cairo-dev python3-ecdsa python3-geoip2 python3-geojson
  python3-graphene-sqlalchemy python3-icalendar python3-maxminddb python3-mpltoolkits.basemap python3-pyproj python3-pyshp python3-requests-file python3-rule-engine
  python3-singledispatch python3-smoke-zephyr xdg-dbus-proxy xdg-desktop-portal xdg-desktop-portal-gtk zenity-common
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libmbedtls libmhash2
Suggested packages:
  libmbedtls-dev mcrpy
The following NEW packages will be installed:
  libmbedtls libmhash2 steghide
0 upgraded, 3 newly installed, 0 to remove and 1311 not upgraded.
Need to get 311 kB of archives.
After this operation, 907 kB of additional disk space will be used.
Get:1 http://ftp.harukasan.org/kali kali-rolling/main amd64 libmbedtls amd64 2.5.8-7 [72.6 kB]
Get:2 http://ftp.harukasan.org/kali kali-rolling/main amd64 libmhash2 amd64 0.9.9.9-9 [94.2 kB]
Get:3 http://ftp.harukasan.org/kali kali-rolling/main amd64 steghide amd64 0.5.1-15 [144 kB]
Fetched 311 kB in 3s (112 kB/s)
Selecting previously unselected package libmbedtls4.
(Reading database ... 276106 files and directories currently installed.)
Preparing to unpack .../libmbedtls4_2.5.8-7_amd64.deb ...
Unpacking libmbedtls4 (2.5.8-7) ...
Selecting previously unselected package libmhash2:amd64.
Preparing to unpack .../libmhash2_0.9.9.9-9_amd64.deb ...
Unpacking libmhash2:amd64 (0.9.9.9-9) ...
Selecting previously unselected package steghide.
Preparing to unpack .../steghide_0.5.1-15_amd64.deb ...
Unpacking steghide (0.5.1-15) ...
Setting up libmbedtls4 (2.5.8-7) ...
Setting up libmhash2:amd64 (0.9.9.9-9) ...
Setting up steghide (0.5.1-15) ...
```

Verifying installation

```
(root@kali)-[~]
# which steghide
/usr/bin/steghide
```

Steghide help: command will show us all the options that Steghide offers us.

```
(root@kali)-[~]
# steghide --help
steghide version 0.5.1

the first argument must be one of the following:
  embed, --embed          embed data
  extract, --extract      extract data
  info, --info            display information about a cover- or stego-file
                        info <filename> display information about <filename>
  encinfo, --encinfo      display a list of supported encryption algorithms
  version, --version      display version information
  license, --license      display steghide's license
  help, --help            display this usage information

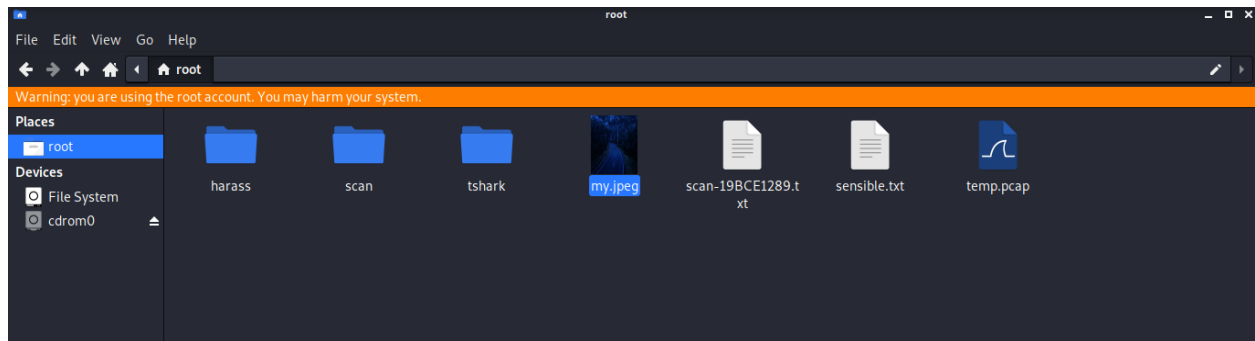
embedding options:
  -ef, --embedfile        select file to be embedded
                        -ef <filename> embed the file <filename>
  -cf, --coverfile        select cover-file
                        -cf <filename> embed into the file <filename>
  -p, --passphrase        specify passphrase
                        -p <passphrase> use <passphrase> to embed data
  -sf, --stegofile        select stego file
                        -sf <filename> write result to <filename> instead of cover-file
  -e, --encryption        select encryption parameters
                        -e <a>[<m>]|<m>[<a>] specify an encryption algorithm and/or mode
  -e none                 do not encrypt data before embedding
  -z, --compress          compress data before embedding (default)
                        -z <l> using level <l> (1 best speed...9 best compression)
  -Z, --dontcompress      do not compress data before embedding
  -K, --nochecksum        do not embed crc32 checksum of embedded data
  -N, --dontembedname     do not embed the name of the original file
  -f, --force             overwrite existing files
  -q, --quiet             suppress information messages
  -v, --verbose           display detailed information
```

Create a txt file in root folder

```
(root@kali)-[~]
# nano sensible.txt
```

```
File Actions Edit View Help
GNU nano 5.4 sensible.txt
My accno is 32456782
```

Save a jpeg image in root folder



```
(root@kali)-[~]
# ls
harass  my.jpeg  scan  scan-19BCE1289.txt  sensible.txt  temp.pcap  tshark
```

Embedding data in the image:

We hide the data in the image using the Steghide so that only the person who accepts it can read it. Therefore, we created a text file named “sensible.txt”, in which we wrote our confidential data and images. JPEG is the file in which we are embedding our data.

```
(root@kali)-[~]
# steghide embed -ef sensible.txt -cf my.jpeg
Enter passphrase:
Re-Enter passphrase:
embedding "sensible.txt" in "my.jpeg" ... done
```

Here, ef and cf are termed as embedded files and cover files, respectively.

Let's see what this command is doing:

Steghide – Program Name

Embed – this is the command

-cf – This flag is for the cover file (the file used to embed the data)

filename – this is the name of the cover file

-ef – This flag is for the embed file (the file that will be embedded)

Filename – This is the name of the embedded file

Extraction of Data From Image Via Steghide:

Using Steghide adds an extra layer of security by allowing us to use a password for it. As long as you know the passphrase, it is quite easy to extract data from the image.

```
(root@kali)~# steghide extract -sf my.jpeg
Enter passphrase:
the file "sensible.txt" does already exist. overwrite ? (y/n) y
wrote extracted data to "sensible.txt".
```

Password Protect Files:

Now, we can also extract files using the following command. This command is different in that it specifies a password in the command itself, therefore, we do not need to specify it separately.

```
(root@kali)~# steghide embed -ef sensible.txt -cf my.jpeg -p 1289
embedding "sensible.txt" in "my.jpeg" ... done
```

```
(root@kali)~# sudo steghide extract -sf my.jpeg -p 1289
the file "sensible.txt" does already exist. overwrite ? (y/n) y
wrote extracted data to "sensible.txt".
```

Retrieve Information of Embedded File:

If we have an image in which the data is suspected to be hidden and if so, what algorithm is used to encrypt the data in the file?

```
(root@kali)~# steghide info my.jpeg
"my.jpeg":
  format: jpeg
  capacity: 19.6 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase:
  embedded file "sensible.txt":
    size: 21.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes
```

Verbose Mode

To obtain every information of a file during extraction, we can use verbose mode. The verbose mode gives you detailed information.

```
(root@kali)~# steghide embed -v -ef sensible.txt -cf my.jpeg
Enter passphrase:
Re-Enter passphrase:
reading secret file "sensible.txt" ... done
reading cover file "my.jpeg" ... done
creating the graph... 113 sample values, 344 vertices, 54287 edges
executing Static Minimum Degree Construction Heuristic... 100.0% (1.0) done
```

Encrypting Algorithms:

We can encrypt the data we are hiding using encryption techniques.

```
(root@kali)~# steghide embed -ef sensible.txt -cf my.jpeg -e des
Enter passphrase:
Re-Enter passphrase:
embedding "sensible.txt" in "my.jpeg" ... done
```

Hiding text file under text file

Creating a new text file

```
(root@kali)~# nano newtext.txt
```

```
File Actions Edit View Help
GNU nano 5.4 newtext.txt
hello frends how are you this is a text file
```

This command encodes the message inside newtext.txt and saves the resulting file that contains the message in newtext1.txt.

```
(root@kali)-[~]
# stegsnow -C -m 'secreat message' newtext.txt newtext1.txt
Compressed by 45.00%
Message exceeded available space by approximately 450.00%.
An extra 2 lines were added.
```

Checking newtext1 file

```
(root@kali)-[~]
# nano newtext1.txt
```

```
File Actions Edit View Help
GNU nano 5.4 newtext1.txt
hello frends how are you this is a text file
```

Checking using stegsnow

```
(root@kali)-[~]
# stegsnow -C newtext1.txt
secreat message
```

Creating sensitive1 file

```
(root@kali)-[~]
# nano sensitive1.txt
```

```
File Actions Edit View Help
GNU nano 5.4 sensitive1.txt *
this is a hidden message
```

Hiding text file under a layer text file

```
(root@kali)-[~]
# stegsnow -C -f sensitive1.txt newtext.txt newtext2.txt
Compressed by 2681212801411272192.00%
Message exceeded available space by approximately 6291.67%.
An extra 26 lines were added.
```

Encode newtext2 file inside org_sensitive file

```
(root@kali)-[~]  
# stegsnow -C newtext2.txt org_sensitive.txt
```

Checking org_sensitive file

```
(root@kali)-[~]  
# nano org_sensitive.txt
```

```
File Actions Edit View Help  
GNU nano 5.4 org_sensitive.txt  
this is a hidden message
```

Encoding using password for keeping file secure

```
(root@kali)-[~]  
# stegsnow -C -p "1234" newtext1.txt org_sensitive.txt
```

Checking org_sensitive file

```
(root@kali)-[~]  
# nano org_sensitive.txt
```

```
File Actions Edit View Help  
GNU nano 5.4 org_sensitive.txt  
bi  
elc-o oeoeo
```