



CONCEALING TEXT AND EXECUTABLE FILES IN JPG FORMAT FILES WITH A STEGANOGRAPHIC COMMAND-LINE TOOL IN THE OPERATING SYSTEM KALI LINUX

Petar Kr. Boyanov

COMMUNICATION AND COMPUTER TECHNOLOGIES, FACULTY OF TECHNICAL SCIENCES, KONSTANTIN PRESLOVSKY UNIVERSITY OF SHUMEN, SHUMEN 9712, 115, UNIVERSITETSKA STR., E-MAIL: petar.boyanov@shu.bg

ABSTRACT: *In this scientific paper concealing text and executable files in JPG format files with a steganographic command-line tool in the Kali Linux operating system is presented. Among the myriad of steganographic tools, Steghide has gained prominence for its robust features and ease of use. Primarily focused on embedding and extracting secret messages within image files, the software tool Steghide enables users to encode text data within JPEG and BMP image formats while maintaining data integrity and minimizing visual alterations. This paper provides an in-depth examination of Steghide's functionality, workflow, and security considerations in the context of digital steganography. It is also discussed the algorithms underpinning Steghide's operations, its strengths, limitations and common use cases.*

KEY WORDS: *Algorithms, Integrity, BMP, JPG, Crack, Linux, LSB, Kali, Privacy, Python3, Security, Steganalysis, Steganography, Stegcracker, Steghide, Wordlists.*

1. Introduction

Unlike cryptography, which obfuscates the content of a message, steganography aims to hide the very existence of the message within a non-suspicious file. With the expansion of digital communication, steganography has adapted to encode data within various digital mediums, including images, audio, and video files. Images, in particular, provide a versatile and visually inconspicuous carrier for hidden messages, given their large amounts of redundant data [1,4,5,6,8,15,19,25,35,39].

Steghide is an open-source tool [4,24,27,32,33,38] that facilitates embedding, extracting, and manipulating hidden messages within images and audio files. It was developed to provide users with a simple yet powerful interface for encoding text or binary data into digital media without introducing noticeable alterations. Steghide uses strong encryption algorithms to ensure that

the embedded data remains secure, even if detected. Its capability to work with various image formats, including BMP and JPEG, further enhances its utility in steganographic applications [1,2,4,10,12,13,24,27,32,33,38,41].

The software program Steghide employs the Least Significant Bit (LSB) [4,24,27,32,33,38] method, implementing techniques in image-based steganography. In the LSB method, Steghide modifies the least significant bits of pixels to encode secret data. Altering these bits has a minimal effect on the overall appearance of the image, making it difficult to visually detect the hidden message [35,36,37,38].

Steghide takes the following steps in the embedding process [40,41,42]:

1. Steghide processes the cover image to analyze its bit distribution and calculates the potential capacity available for embedding data.
2. Before embedding, the secret message is encrypted using a secure encryption algorithm (usually AES-128 or 256-bit) to protect the contents of the hidden message, even if detected.
3. The encrypted data is then embedded in the LSBs of the cover image. Since only the least significant bits are altered, visual distortions are minimized, preserving the image's integrity.
4. Steghide incorporates error-detection codes within the embedding process to ensure data integrity during extraction.

Steghide represents a significant tool in the field of digital steganography. Its flexibility in data hiding and extracting, along with its encryption capabilities, makes it a valuable asset in maintaining privacy and secure communication in the digital age. In this regards, the software tool Steghide has a wide range of applications in fields where data confidentiality and covert communication are essential [2,3,4,9,10,11,12,34,35,36,38,42,43].

The conducted experiments in this scientific paper that aim to embed, capture, crack and decode important and confidential information without the host's permission is considered as a crime and, if proven, is punishable to the full extent of the law of the respective country [5,6]. Everything illustrated and explained in this scientific paper is for research work and educational purposes and the author is not responsible in cases of abuse.

2. Related work

These scientific works [1,2,4,5,6,10,11,12,19,20,25,32,33] collectively explore various aspects of steganography and steganalysis, including methods for enhancing the security of tools like Steghide, detecting hidden data with Stegcracker, and examining the strengths and limitations of LSB techniques in image-based data hiding. Each study contributes to a deeper understanding of both embedding and detection strategies in the field of steganography [1,10,16,23,27,29,35,36,37,38,40,41,42].

These references [4,5,6,7,8,9,13,15,21,24,26,28,39,43] an extensive perspective on steganography, steganalysis, and data hiding techniques and cover foundational theories, practical applications, and detection methodologies essential for understanding and analyzing the effectiveness of Steghide and similar tools.

The concealing text and executable files is also used in application of electronic platforms [30], various types of instrumental equipment for cyberattack prevention [23], specific models for accessing information resources in a secure environment and other technologies [22], net model of command and control system [17], building data center system for defense and security [16], designing and implementation of software-defined systems [18], information exchange management in multimodule multi-position security systems [14], applications of Artificial Intelligence in e-Learning [31], information systems for crisis prevention [34] and designing of stream ciphers based on random feedback shift registers [3].

3. Experiment

The scientific experiments in this paper in a specialized computer network laboratory in the Faculty of Technical Sciences of the Konstantin Preslavsky University of Shumen are made [5,6].

The used operating system is Kali Linux (Linux pesho 6.0.0-kali6-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.0.12-1kali1 x86_64 GNU/Linux). Steghide is available on major platforms, including Linux, Windows, and macOS [5,6].

The installation typically requires the following commands for Linux-based systems: „which steghide“ and „sudo apt-get install steghide“. This is shown in fig. 1.

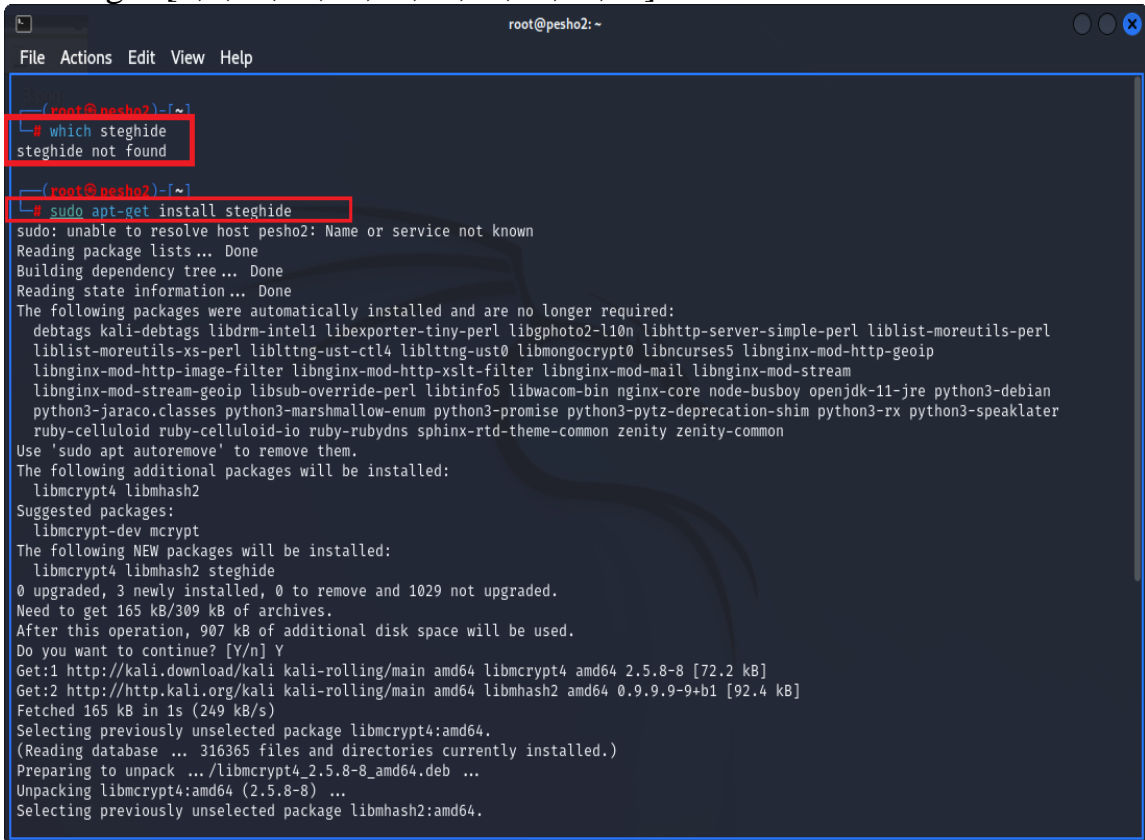
The next task a text file is to be created that contains the following text: “Warning! This is a test file!” and an executable file that sends ping requests to the journal’s site - jsar.ftn.shu.bg.

Fig. 2 illustrates the creation of text and executable files. Practically Steghide is designed to work with four primary file formats: WAV, AU, BMP and JPEG files. In this connection, two image files (1.jpg and 2.jpg) for the scientific research are used.

The software tool Steghide is designed to work with two primary image formats [1,2,10,11,14,15,32,33,40,41,42,43]:

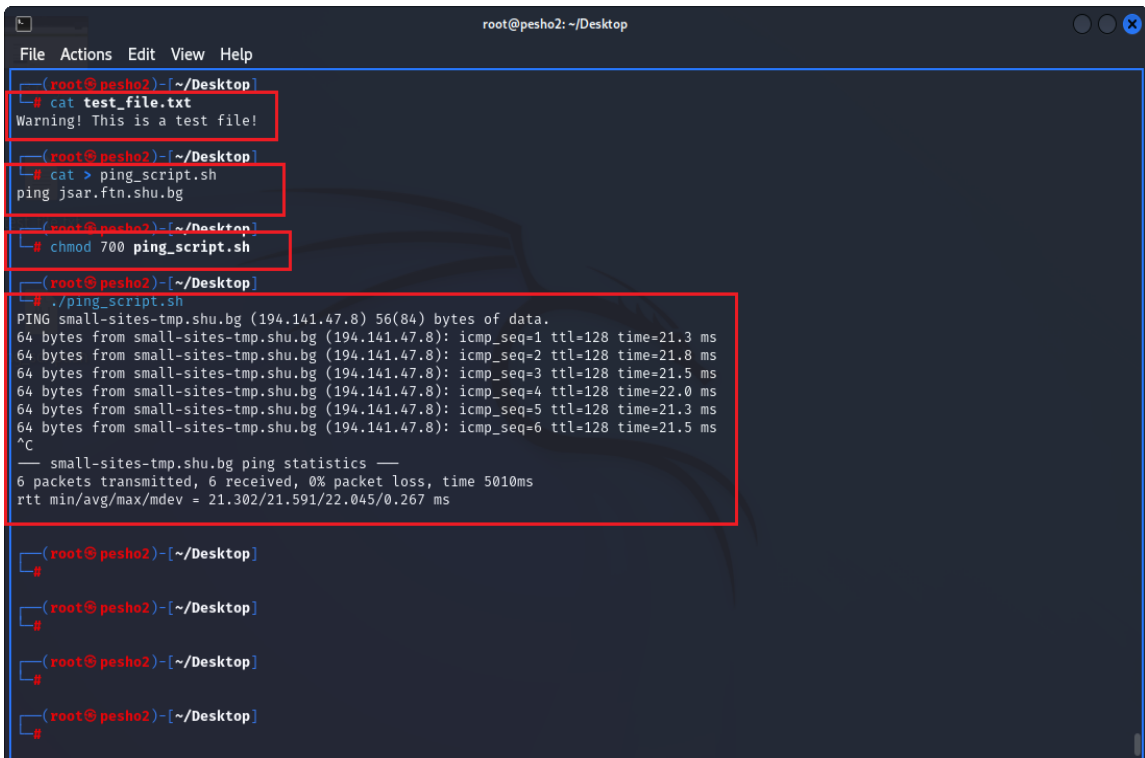
- **Bitmap (BMP):** This format supports uncompressed image data, which allows for more flexibility in embedding messages without significantly impacting file size [4,24,27,32,33,38,40,41].
- **JPEG:** This compressed format is commonly used for photographs. While challenging for traditional LSB substitution, Steghide uses

advanced embedding techniques to maintain quality and reduce file size changes [1,4,20,21,24,27,32,33,38,42,43].



```
root@pesho2: ~  
File Actions Edit View Help  
root@pesho2:~# which steghide  
steghide not found  
root@pesho2:~# sudo apt-get install steghide  
sudo: unable to resolve host pesho2: Name or service not known  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following packages were automatically installed and are no longer required:  
debtags kali-debtags libdrm-intel1 libexporter-tiny-perl libgphoto2-l10n libhttp-server-simple-perl liblist-moreutils-perl  
liblist-moreutils-xs-perl liblttng-ust-ctl4 liblttng-ust0 libmongocrypt0 libncurses5 libnginx-mod-http-geoip  
libnginx-mod-http-image-filter libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream  
libnginx-mod-stream-geoip libsub-override-perl libtinfo5 libwacom-bin nginx-core node-busboy openjdk-11-jre python3-debian  
python3-jaraco.classes python3-marshmallow-enum python3-promise python3-pytz-deprecation-shim python3-rx python3-speaklater  
ruby-celluloid ruby-celluloid-io ruby-rubydns sphinx-rtd-theme-common zenity zenity-common  
Use 'sudo apt autoremove' to remove them.  
The following additional packages will be installed:  
  libcrypt4 libmhash2  
Suggested packages:  
  libcrypt-dev mcrypt  
The following NEW packages will be installed:  
  libcrypt4 libmhash2 steghide  
0 upgraded, 3 newly installed, 0 to remove and 1029 not upgraded.  
Need to get 165 kB/309 kB of archives.  
After this operation, 907 kB of additional disk space will be used.  
Do you want to continue? [Y/n] Y  
Get:1 http://kali.download/kali kali-rolling/main amd64 libcrypt4 amd64 2.5.8-8 [72.2 kB]  
Get:2 http://http.kali.org/kali kali-rolling/main amd64 libmhash2 amd64 0.9.9.9-9+b1 [92.4 kB]  
Fetched 165 kB in 1s (249 kB/s)  
Selecting previously unselected package libcrypt4:amd64.  
(Reading database ... 316365 files and directories currently installed.)  
Preparing to unpack .../libcrypt4_2.5.8-8_amd64.deb ...  
Unpacking libcrypt4:amd64 (2.5.8-8) ...  
Selecting previously unselected package libmhash2:amd64.
```

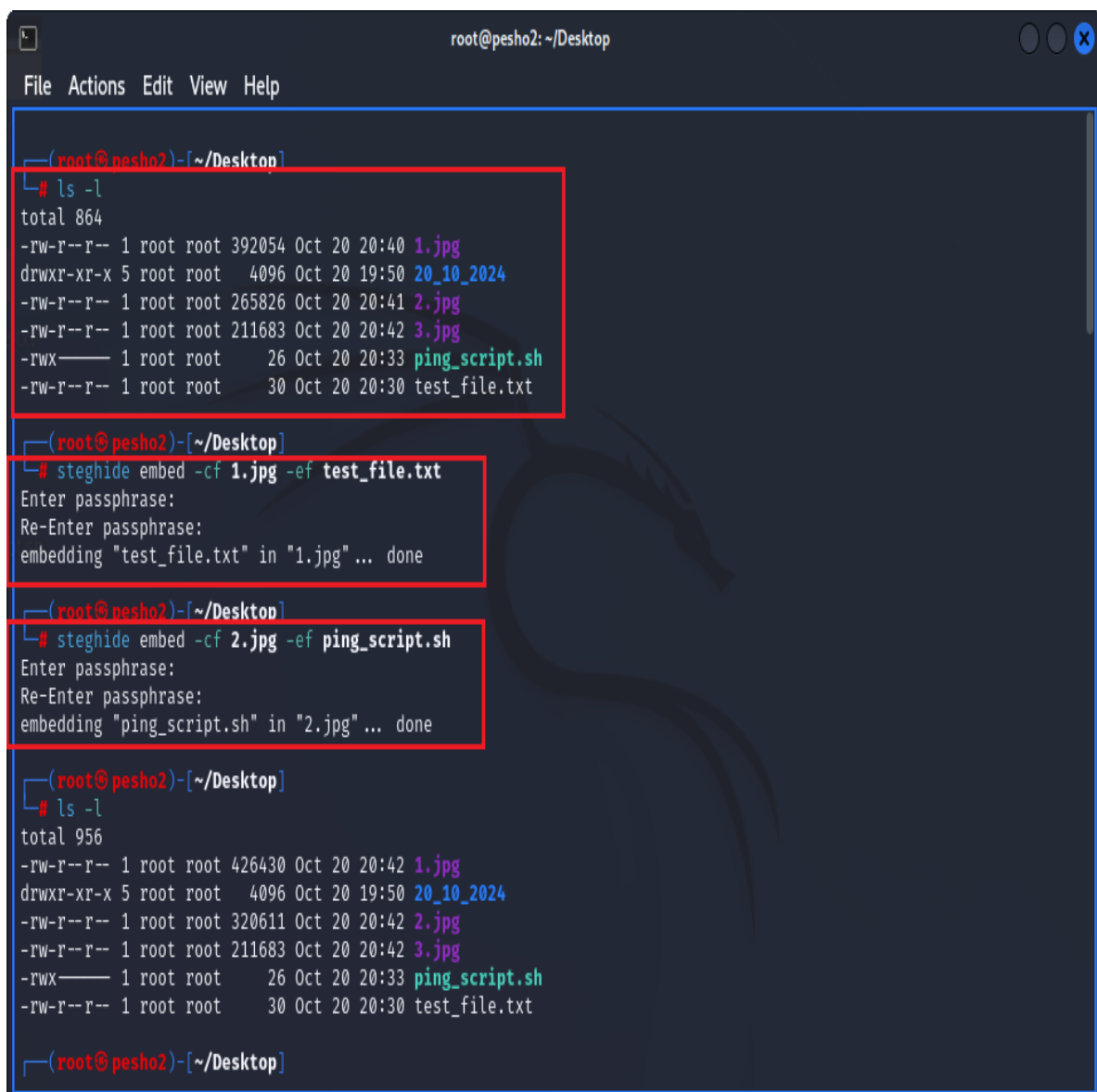
Fig. 1. The process of installation of Steghide



```
root@pesho2: ~/Desktop  
File Actions Edit View Help  
root@pesho2:~/Desktop# cat test_file.txt  
Warning! This is a test file!  
root@pesho2:~/Desktop# cat > ping_script.sh  
ping jsar.ftn.shu.bg  
root@pesho2:~/Desktop# chmod 700 ping_script.sh  
root@pesho2:~/Desktop# ./ping_script.sh  
PING small-sites-tmp.shu.bg (194.141.47.8) 56(84) bytes of data.  
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=1 ttl=128 time=21.3 ms  
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=2 ttl=128 time=21.8 ms  
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=3 ttl=128 time=21.5 ms  
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=4 ttl=128 time=22.0 ms  
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=5 ttl=128 time=21.3 ms  
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=6 ttl=128 time=21.5 ms  
^C  
  -- small-sites-tmp.shu.bg ping statistics --  
6 packets transmitted, 6 received, 0% packet loss, time 5010ms  
rtt min/avg/max/mdev = 21.302/21.591/22.045/0.267 ms
```

Fig. 2. The creation of text and executable files

The process of embedding a message using Steghide involves selecting a cover image and a secret message file. The syntax of the first command for embedding a message is as follows: “steghide embed -cf 1.jpg -ef test_file.txt”. The name of the text file is “test_file.txt” and the name of the executable file is “ping_script.sh”. The syntax of the second command for embedding a message is as follows: “steghide embed -cf 2.jpg -ef ping_script.sh”. The software program Steghide [1,4,20,21,24,27,32,33,38,42,43].sets passphrase that will be required for extracting the hidden message. All these steps for embedding a message in fig. 3 are shown. The command “ls -l” (fig. 3) lists detailed long information for all files used the scientific work.



```
root@pesho2: ~/Desktop
File Actions Edit View Help

root@pesho2)~[/Desktop]
# ls -l
total 864
-rw-r--r-- 1 root root 392054 Oct 20 20:40 1.jpg
drwxr-xr-x 5 root root 4096 Oct 20 19:50 20_10_2024
-rw-r--r-- 1 root root 265826 Oct 20 20:41 2.jpg
-rw-r--r-- 1 root root 211683 Oct 20 20:42 3.jpg
-rwx----- 1 root root 26 Oct 20 20:33 ping_script.sh
-rw-r--r-- 1 root root 30 Oct 20 20:30 test_file.txt

root@pesho2)~[/Desktop]
# steghide embed -cf 1.jpg -ef test_file.txt
Enter passphrase:
Re-Enter passphrase:
embedding "test_file.txt" in "1.jpg" ... done

root@pesho2)~[/Desktop]
# steghide embed -cf 2.jpg -ef ping_script.sh
Enter passphrase:
Re-Enter passphrase:
embedding "ping_script.sh" in "2.jpg" ... done

root@pesho2)~[/Desktop]
# ls -l
total 956
-rw-r--r-- 1 root root 426430 Oct 20 20:42 1.jpg
drwxr-xr-x 5 root root 4096 Oct 20 19:50 20_10_2024
-rw-r--r-- 1 root root 320611 Oct 20 20:42 2.jpg
-rw-r--r-- 1 root root 211683 Oct 20 20:42 3.jpg
-rwx----- 1 root root 26 Oct 20 20:33 ping_script.sh
-rw-r--r-- 1 root root 30 Oct 20 20:30 test_file.txt

root@pesho2)~[/Desktop]
```

Fig. 3. The process of embedding a message in text file

The two image files used for the scientific experiments in Fig. 4 and 5 are presented.

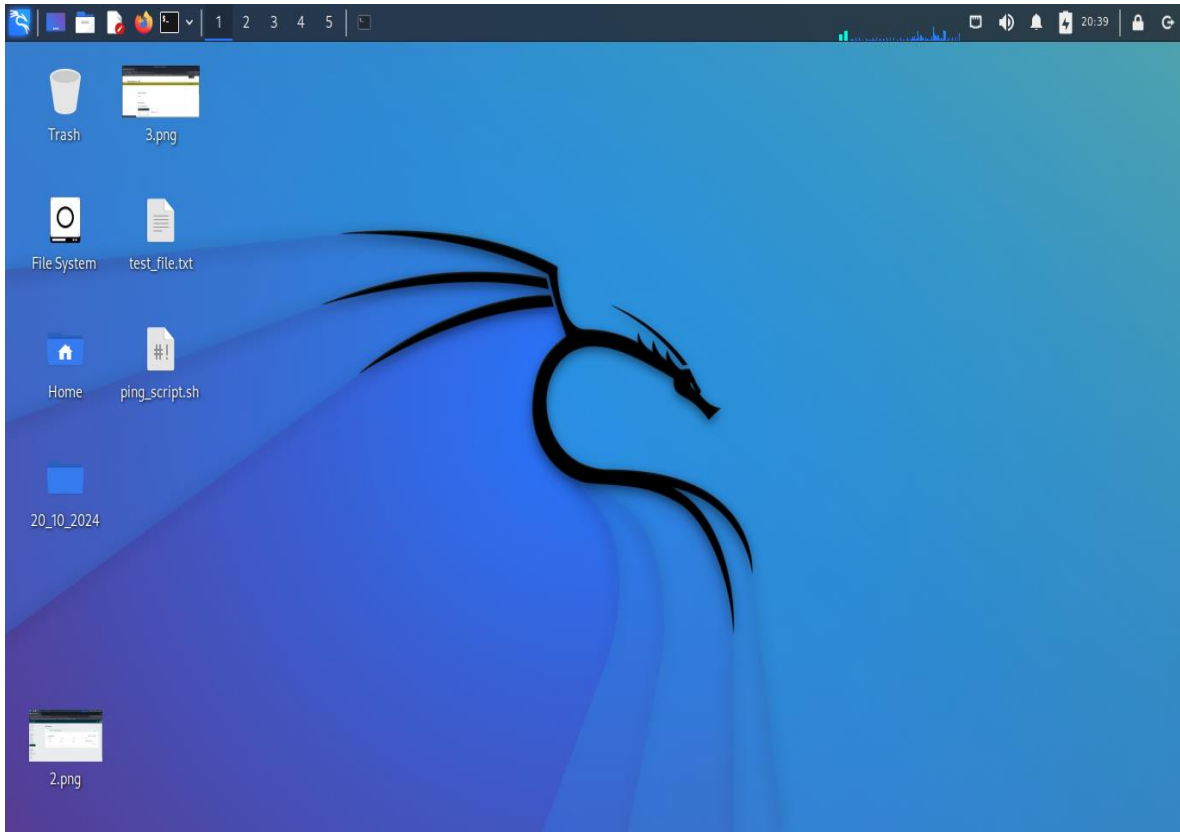


Fig. 4. The first image file (1.jpg)

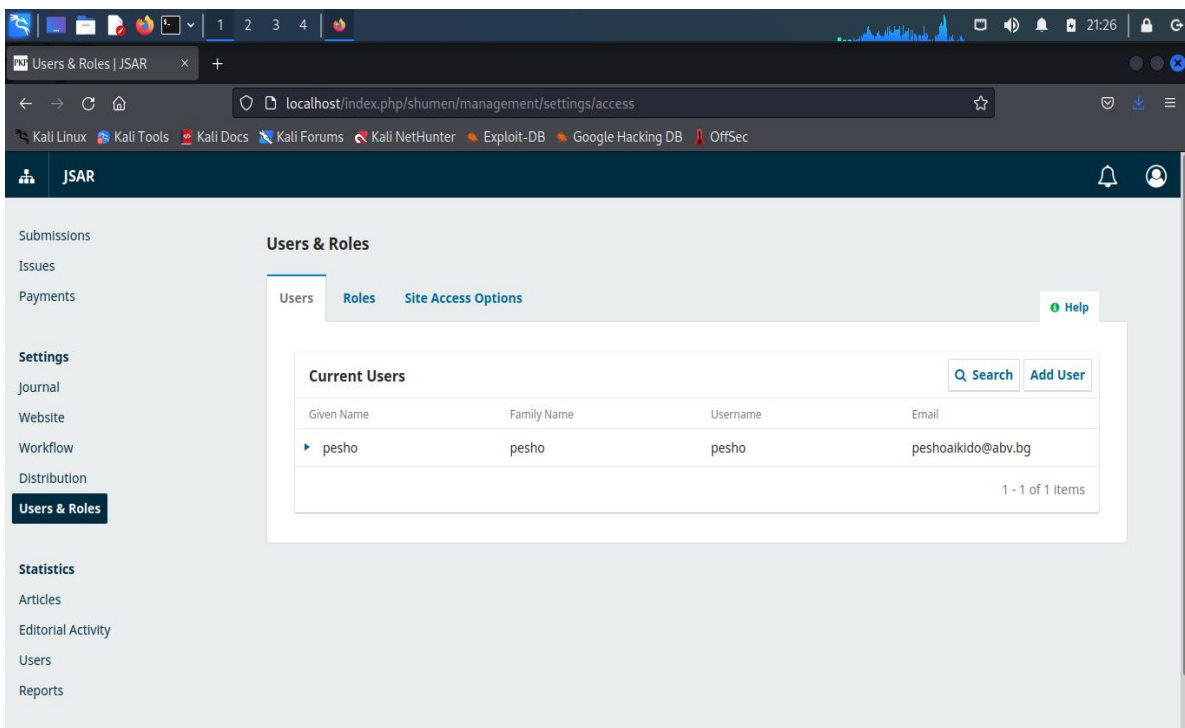
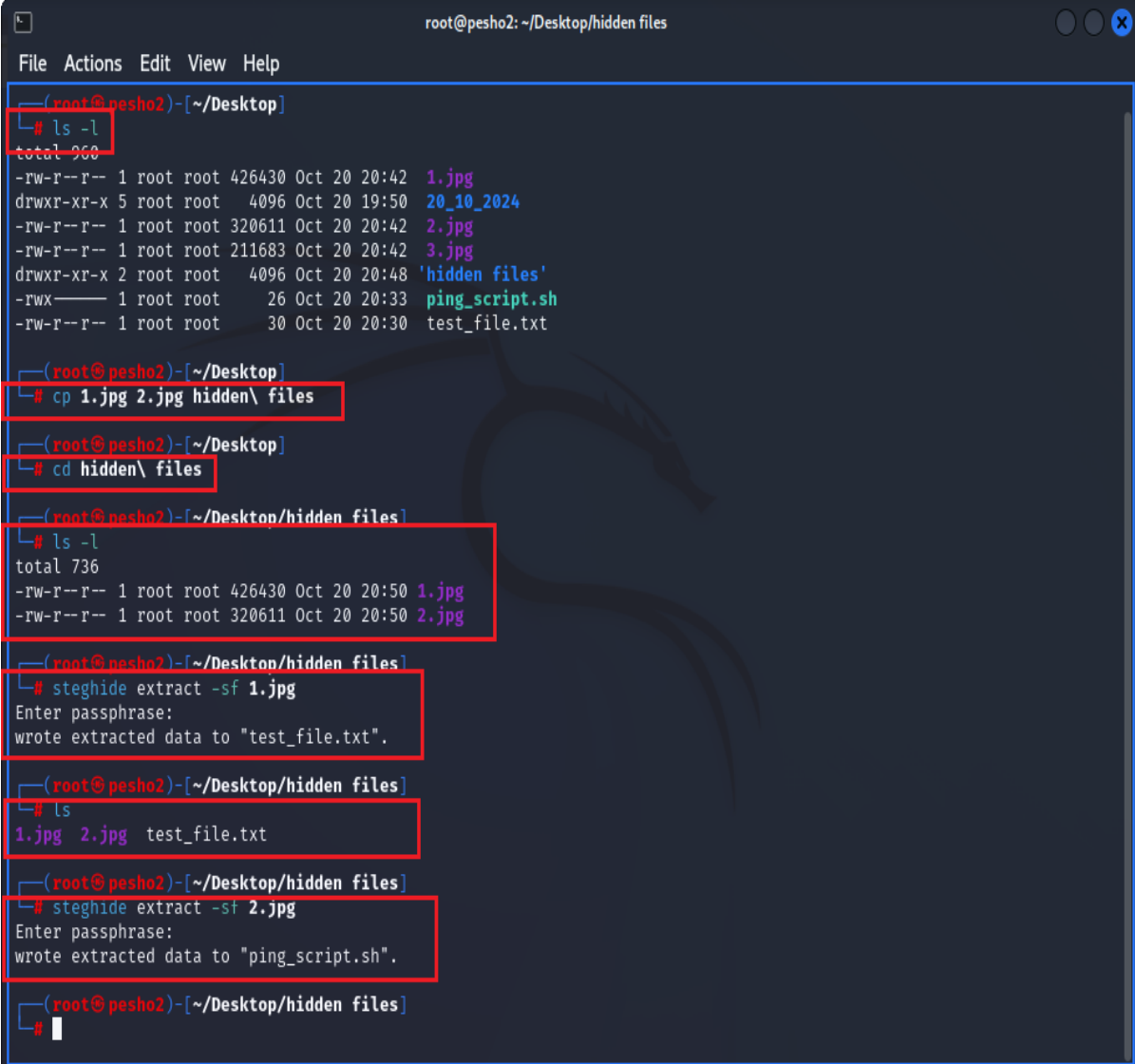


Fig. 5. The second image file (2.jpg)

For clarity, both image files 1.jpg and 2.jpg to another directory (~/Desktop/hidden files) have been copied. The extracting of the hidden files

from both image files (1.jpg and 2.jpg) requires the following two commands to be written in the terminal: “steghide extract -sf 1.jpg” and “steghide extract -sf 2.jpg” (shown in fig. 6).



```
root@pesho2: ~/Desktop/hidden files
File Actions Edit View Help
(root@pesho2)~/Desktop
# ls -l
total 260
-rw-r--r-- 1 root root 426430 Oct 20 20:42 1.jpg
drwxr-xr-x 5 root root 4096 Oct 20 19:50 20_10_2024
-rw-r--r-- 1 root root 320611 Oct 20 20:42 2.jpg
-rw-r--r-- 1 root root 211683 Oct 20 20:42 3.jpg
drwxr-xr-x 2 root root 4096 Oct 20 20:48 'hidden files'
-rwx----- 1 root root 26 Oct 20 20:33 ping_script.sh
-rw-r--r-- 1 root root 30 Oct 20 20:30 test_file.txt

(root@pesho2)~/Desktop
# cp 1.jpg 2.jpg hidden\ files

(root@pesho2)~/Desktop
# cd hidden\ files

(root@pesho2)~/Desktop/hidden files
# ls -l
total 736
-rw-r--r-- 1 root root 426430 Oct 20 20:50 1.jpg
-rw-r--r-- 1 root root 320611 Oct 20 20:50 2.jpg

(root@pesho2)~/Desktop/hidden files
# steghide extract -sf 1.jpg
Enter passphrase:
wrote extracted data to "test_file.txt".

(root@pesho2)~/Desktop/hidden files
# ls
1.jpg 2.jpg test_file.txt

(root@pesho2)~/Desktop/hidden files
# steghide extract -sf 2.jpg
Enter passphrase:
wrote extracted data to "ping_script.sh".

(root@pesho2)~/Desktop/hidden files
#
```

Fig. 6. Extracting the hidden files from both image files (1.jpg and 2.jpg)

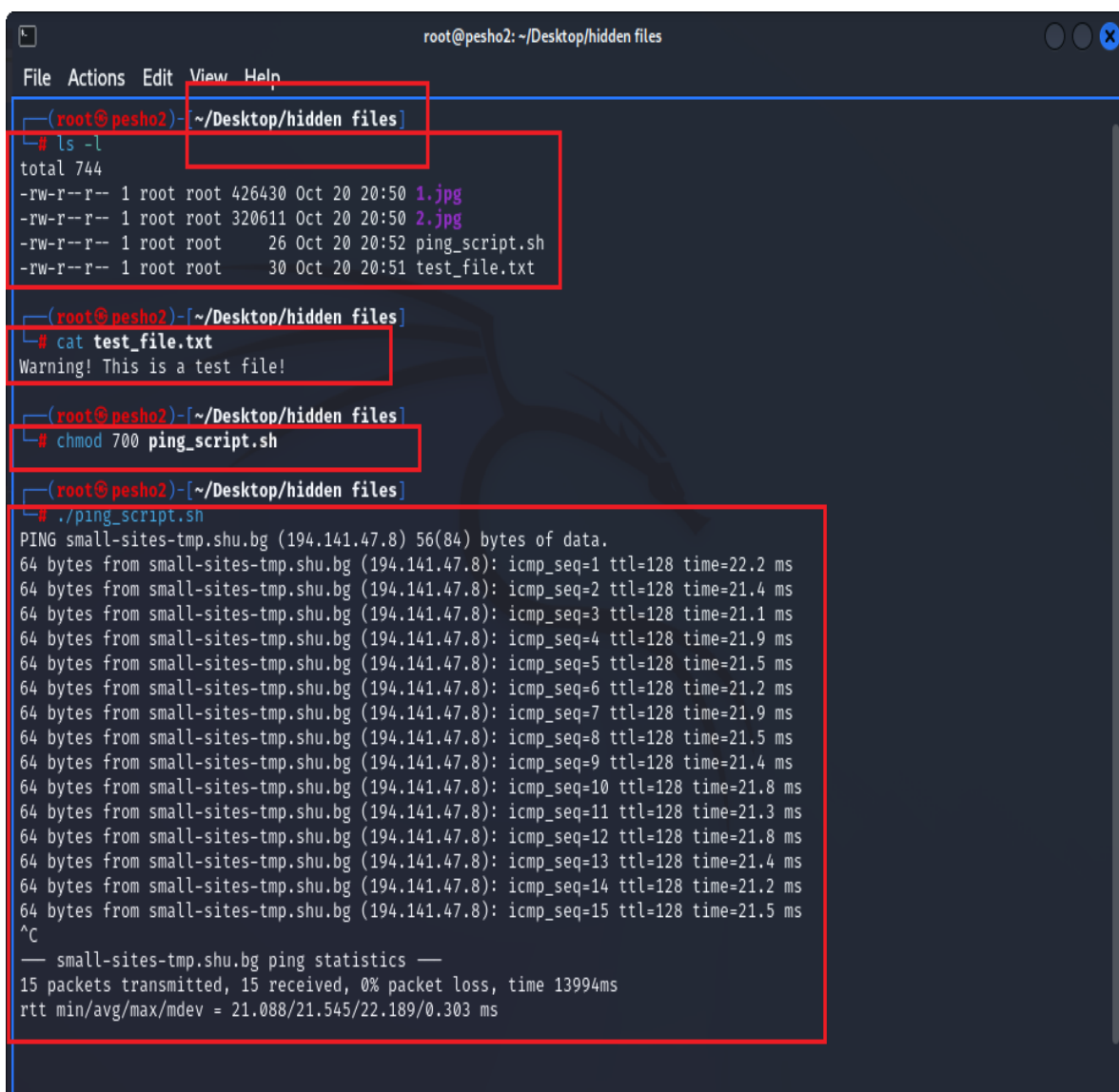
The used encryption mechanism of the program Steghide is integral to its security model. By using AES encryption, Steghide ensures that even if someone suspects the presence of a hidden message, deciphering it without the correct password is practically infeasible. However, the users of the software program Steghide should always use complex, unique passwords to prevent brute-force attacks [1,4,5,8,9,24,27,28,29,32,33,38].

Data integrity is maintained using built-in error-detection codes, which allow Steghide to verify the accuracy of the extracted user data. This feature is especially beneficial when embedding messages in compressed formats like JPEG, where slight distortions can occur [1,3,4,10,20,21,23,24,31,32,34,35].

Steghide's primary steganographic defense is its use of LSB substitution, which produces minimal visual changes in the cover image. However, specialized steganalysis tools, especially those examining statistical anomalies in LSB patterns, may potentially detect hidden messages [4,24,27,32,33,38]. Therefore, combining steganography with strong encryption enhances message security, making it difficult for an adversary to interpret the data even if detected [4,5,6,7,23,26,27,29,31,42].

4. Results

Fig. 7 illustrates the copied files in the directory (~/Desktop/hidden files) and the opening the extracted text file (test_file.txt) and the execution of the script (ping_script.sh). This is shown in fig.7.



```
root@pesho2: ~/Desktop/hidden files
File Actions Edit View Help
root@pesho2)~/Desktop/hidden files]
# ls -l
total 744
-rw-r--r-- 1 root root 426430 Oct 20 20:50 1.jpg
-rw-r--r-- 1 root root 320611 Oct 20 20:50 2.jpg
-rw-r--r-- 1 root root 26 Oct 20 20:52 ping_script.sh
-rw-r--r-- 1 root root 30 Oct 20 20:51 test_file.txt

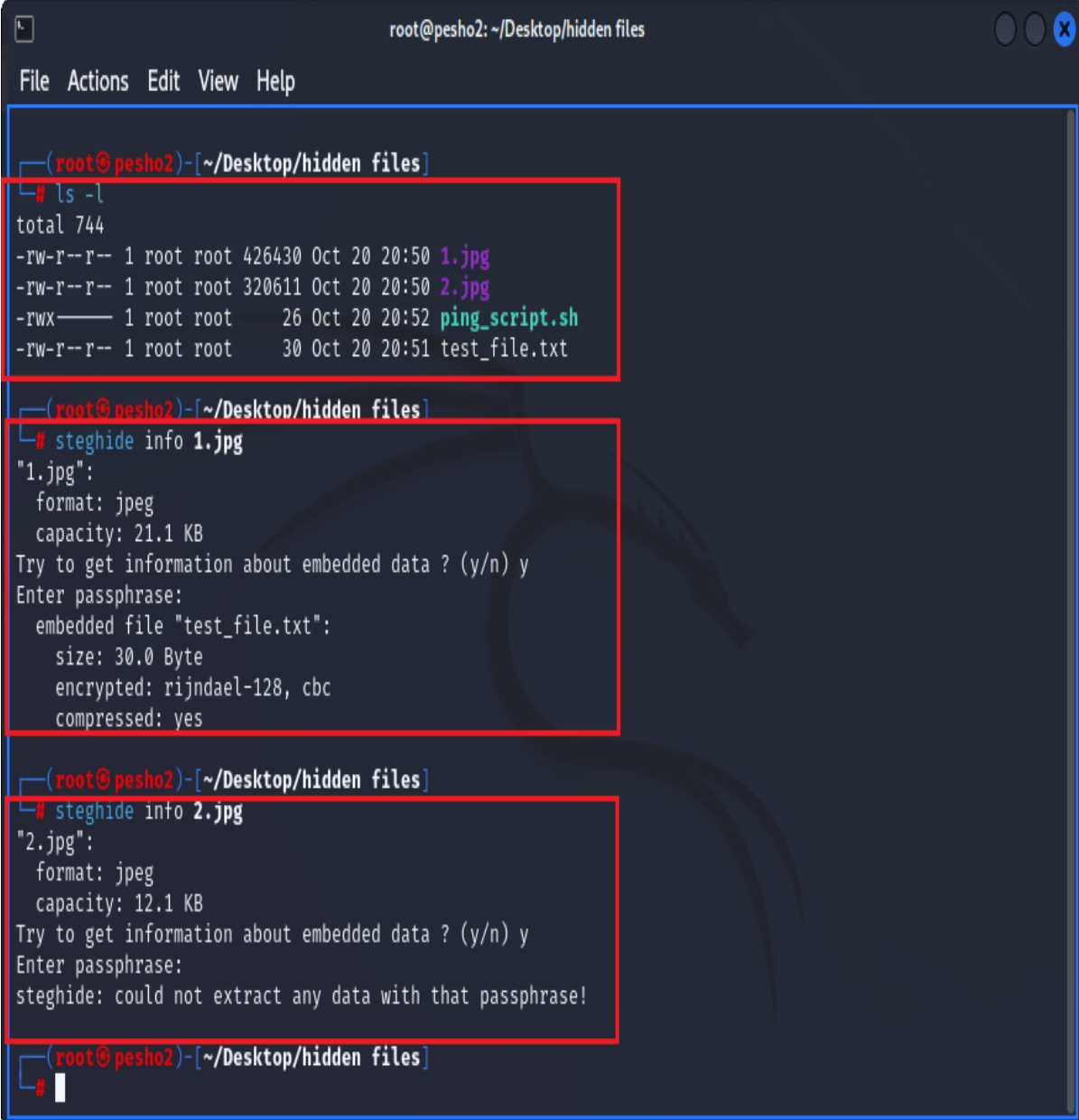
root@pesho2)~/Desktop/hidden files]
# cat test_file.txt
Warning! This is a test file!

root@pesho2)~/Desktop/hidden files]
# chmod 700 ping_script.sh

root@pesho2)~/Desktop/hidden files]
# ./ping_script.sh
PING small-sites-tmp.shu.bg (194.141.47.8) 56(84) bytes of data.
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=1 ttl=128 time=22.2 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=2 ttl=128 time=21.4 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=3 ttl=128 time=21.1 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=4 ttl=128 time=21.9 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=5 ttl=128 time=21.5 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=6 ttl=128 time=21.2 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=7 ttl=128 time=21.9 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=8 ttl=128 time=21.5 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=9 ttl=128 time=21.4 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=10 ttl=128 time=21.8 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=11 ttl=128 time=21.3 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=12 ttl=128 time=21.8 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=13 ttl=128 time=21.4 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=14 ttl=128 time=21.2 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=15 ttl=128 time=21.5 ms
^C
— small-sites-tmp.shu.bg ping statistics —
15 packets transmitted, 15 received, 0% packet loss, time 13994ms
rtt min/avg/max/mdev = 21.088/21.545/22.189/0.303 ms
```

Fig. 7. listing the directory (~/Desktop/hidden files) and executing the both extracted files

Fig. 8 shows basic information about the two image files - 1.jpg and 2.jpg. In case the user wants to get more information about the image file, then he will have to enter the passphrase. When a correct password is entered, the name and extension of the hidden file, its size, encryption method and compression will be displayed (1.jpg). If the user does not know the password for extraction, then no information about the hidden file can be obtained (2.jpg). This is shown in fig. 8.



```
root@pesho2: ~/Desktop/hidden files
File Actions Edit View Help

(root@pesho2)-[~/Desktop/hidden files]
# ls -l
total 744
-rw-r--r-- 1 root root 426430 Oct 20 20:50 1.jpg
-rw-r--r-- 1 root root 320611 Oct 20 20:50 2.jpg
-rwx----- 1 root root 26 Oct 20 20:52 ping_script.sh
-rw-r--r-- 1 root root 30 Oct 20 20:51 test_file.txt

(root@pesho2)-[~/Desktop/hidden files]
# steghide info 1.jpg
"1.jpg":
  format: jpeg
  capacity: 21.1 KB
  Try to get information about embedded data ? (y/n) y
  Enter passphrase:
  embedded file "test_file.txt":
    size: 30.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes

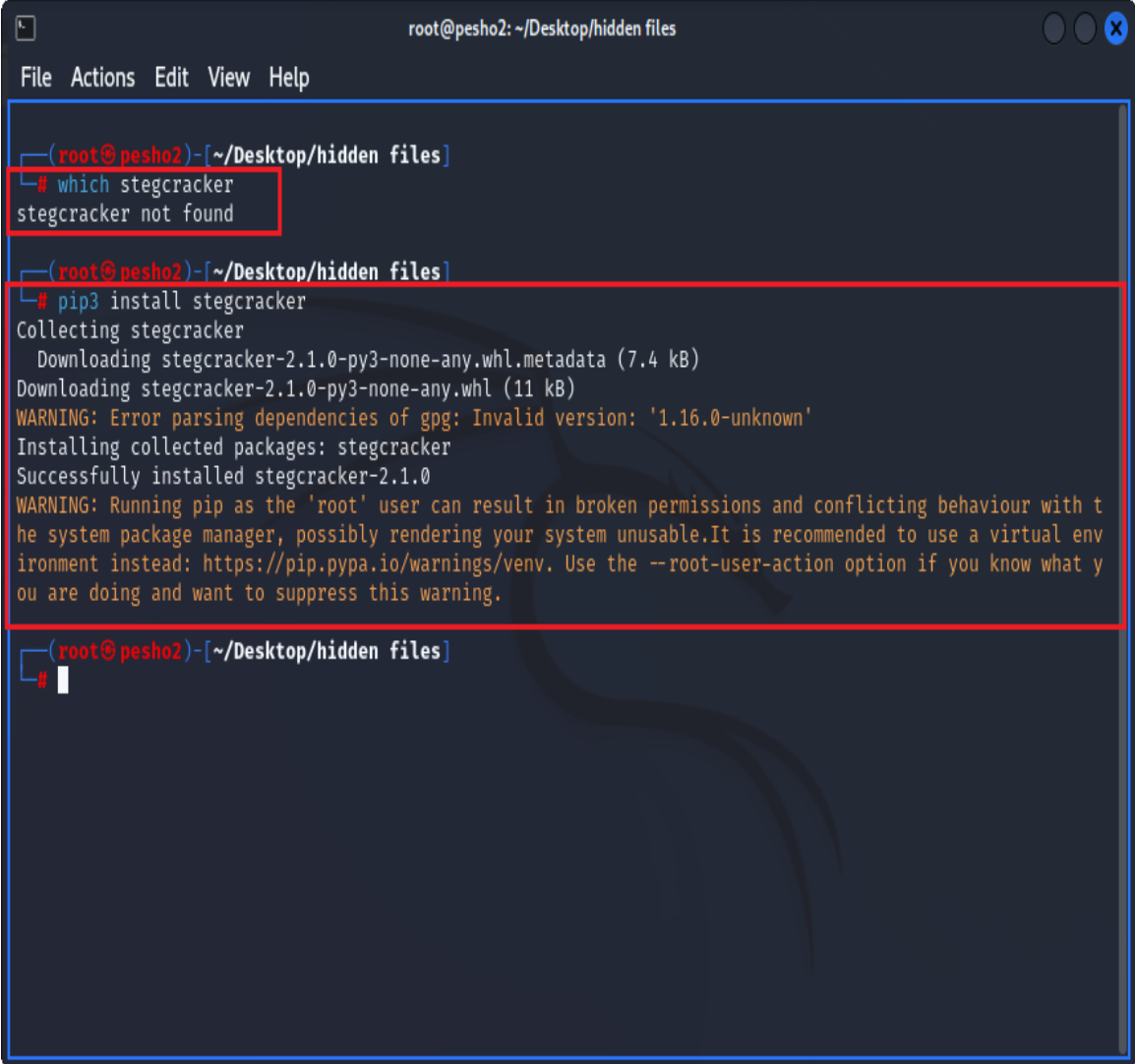
(root@pesho2)-[~/Desktop/hidden files]
# steghide info 2.jpg
"2.jpg":
  format: jpeg
  capacity: 12.1 KB
  Try to get information about embedded data ? (y/n) y
  Enter passphrase:
  steghide: could not extract any data with that passphrase!

(root@pesho2)-[~/Desktop/hidden files]
#
```

Fig. 8. File's information with the command "steghide info"

There is a software program that can crack the passphrase (password). Its name is Stegcracker and it is a command line tool. The installation typically requires the following commands for Linux-based systems: „which stegcracker“

and „sudo pip3 install stegcracker“. This is shown in fig. 9. The installed version of the program is 2.1.0. Before running the application, it is necessary to create a password file. Through this file, the application will try to reveal the password of the image file. The other option is to be used ready-made wordlists. On the Kali Linux operating system, they are located in the directory /usr/share/wordlists [1,4,5,6,20,21,24,27,32,33,38,42,43]. For the purposes of this scientific research, a text file consisting of several passwords is created. The syntax of the used command is: “sudo stegcracker 1.jpg paroli.txt” and “sudo stegcracker 2.jpg paroli.txt”. This is presented in fig. 10 and 11. As can be seen from the achieved results, the application successfully managed to reveal the password for the both image file 1.jpg and 2.jpg with a total of 19 attempts. The first cracked image file “1.jpg” to “1.jpg.out” is written. The second cracked image file “2.jpg” to “2.jpg.out” is also written. After that the user can see the content of the both cracked image files with command “cat”. This is shown in fig. 12.



```
root@pesho2: ~/Desktop/hidden files
File Actions Edit View Help

(root@pesho2)-[~/Desktop/hidden files]
# which stegcracker
stegcracker not found

(root@pesho2)-[~/Desktop/hidden files]
# pip3 install stegcracker
Collecting stegcracker
  Downloading stegcracker-2.1.0-py3-none-any.whl.metadata (7.4 kB)
  Downloading stegcracker-2.1.0-py3-none-any.whl (11 kB)
WARNING: Error parsing dependencies of gpg: Invalid version: '1.16.0-unknown'
Installing collected packages: stegcracker
Successfully installed stegcracker-2.1.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager, possibly rendering your system unusable. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv. Use the --root-user-action option if you know what you are doing and want to suppress this warning.

(root@pesho2)-[~/Desktop/hidden files]
#
```

Fig. 9. The installation of the tool “Stegcracker”

```
File Actions Edit View Help
paroli.
└─(root@pesho)-[~/Desktop]
# steghide info 1.jpg
"1.jpg":
  format: jpeg
  capacity: 21.1 KB
1.jpg.o Try to get information about embedded data ? (y/n) y
Enter passphrase:
steghide: could not extract any data with that passphrase!

└─(root@pesho)-[~/Desktop]
# stegcracker 1.jpg paroli.txt
StegCracker 2.1.0 - (https://github.com/Paradoxis/StegCracker)
Copyright (c) 2024 - Luke Paris (Paradoxis)

StegCracker has been retired following the release of StegSeek, which
will blast through the rockyou.txt wordlist within 1.9 second as opposed
to StegCracker which takes ~5 hours.

StegSeek can be found at: https://github.com/RickdeJager/stegseek

Counting lines in wordlist..
Attacking file '1.jpg' with wordlist 'paroli.txt'..
Successfully cracked file with password: 123
Tried 19 passwords
Your file has been written to: 1.jpg.out
123

└─(root@pesho)-[~/Desktop]
# stegcracker 2.jpg paroli.txt
StegCracker 2.1.0 - (https://github.com/Paradoxis/StegCracker)
Copyright (c) 2024 - Luke Paris (Paradoxis)

StegCracker has been retired following the release of StegSeek, which
will blast through the rockyou.txt wordlist within 1.9 second as opposed
```

Fig. 10. The executed command stegckacker 1.jpg parole.txt

```
File Actions Edit View Help
paroli.
Copyright (c) 2024 - Luke Paris (Paradoxis)

StegCracker has been retired following the release of StegSeek, which
will blast through the rockyou.txt wordlist within 1.9 second as opposed
to StegCracker which takes ~5 hours.

1.jpg.o StegSeek can be found at: https://github.com/RickdeJager/stegseek

Counting lines in wordlist..
Attacking file '1.jpg' with wordlist 'paroli.txt'..
Successfully cracked file with password: 123
Tried 19 passwords
Your file has been written to: 1.jpg.out
123

└─(root@pesho)-[~/Desktop]
# stegcracker 2.jpg paroli.txt
StegCracker 2.1.0 - (https://github.com/Paradoxis/StegCracker)
Copyright (c) 2024 - Luke Paris (Paradoxis)

StegCracker has been retired following the release of StegSeek, which
will blast through the rockyou.txt wordlist within 1.9 second as opposed
to StegCracker which takes ~5 hours.

StegSeek can be found at: https://github.com/RickdeJager/stegseek

Counting lines in wordlist..
Attacking file '2.jpg' with wordlist 'paroli.txt'..
Successfully cracked file with password: 123
Tried 19 passwords
Your file has been written to: 2.jpg.out
123

└─(root@pesho)-[~/Desktop]
#
```

Fig. 11. The executed command stegckacker 2.jpg parole.txt

```

File Actions Edit View Help
root@pesho: ~/Desktop
# ls -l
total 1476
-rw-r--r-- 1 root root 366774 Oct 21 02:01 11.png
-rw-r--r-- 1 root root 366285 Oct 21 02:02 12.png
-rw-r--r-- 1 root root 426430 Oct 21 01:53 1.jpg
-rw-r--r-- 1 root root 30 Oct 21 01:59 1.jpg.out
drwxr-xr-x 6 root root 4096 Oct 20 19:43 20_10_2024
-rw-r--r-- 1 root root 320611 Oct 21 01:54 2.png
-rw-r--r-- 1 root root 26 Oct 21 01:59 2.jpg.out
-rw-r--r-- 1 root root 70 Oct 21 01:40 paroli.txt
-rw-r--r-- 1 root root 998 Jun 3 2022 Toolkit.lnk

root@pesho: ~/Desktop
# cat 1.jpg.out
Warning! This is a test file!

root@pesho: ~/Desktop
# cat 2.jpg.out
ping jsar.ftn.shu.bg
date

root@pesho: ~/Desktop
# chmod 700 2.jpg.out

root@pesho: ~/Desktop
# ./2.jpg.out
PING small-sites-tmp.shu.bg (194.141.47.8) 56(84) bytes of data.
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=1 ttl=128 time=22.1 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=2 ttl=128 time=21.1 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=3 ttl=128 time=21.7 ms
64 bytes from small-sites-tmp.shu.bg (194.141.47.8): icmp_seq=4 ttl=128 time=24.5 ms
^C
--- small-sites-tmp.shu.bg ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms

```

Fig. 12. The content of the both cracked image files - “1.jpg.out” and “2.jpg.out”

As a result of the scientific research done, the following advantages can be deduced [4,24,27,32,33,35,36,38,40,42]:

- The software tool Steghide is straightforward to use, with command-line options that allow for quick embedding and extraction.
- Strong encryption ensures that even if an adversary suspects the presence of hidden data, decrypting it is difficult without the password.
- The use of LSB substitution minimizes visible distortions, preserving the original appearance of the cover image.

As a result of the scientific research done, the following limitations can be deduced [1,2,8,9,11,12,13,39,43]:

- The tool Steghide is primarily restricted to BMP and JPEG formats, limiting its flexibility in some contexts.
- While Steghide is effective against casual inspection, sophisticated steganalysis tools can potentially detect the presence of hidden data.
- Steghide can only embed a limited amount of data, dependent on the size of the cover image.

5. Conclusion

Steghide [1,4,20,21,24,27,32,33,38,42,43] remains a valuable tool in digital steganography, especially for users seeking a combination of simplicity, encryption, and visual integrity. By coupling traditional steganographic

techniques with strong encryption, Steghide demonstrates that simple tools can offer substantial security in the field of information hiding. Steghide represents a significant tool in the field of digital steganography. Its flexibility in data hiding and extracting, along with its encryption capabilities, makes it a valuable asset in maintaining privacy and secure communication in the digital age.

Enhancing Steghide [4,11,29] to support additional file formats, improve embedding capacity, and incorporate advanced resistance against modern steganalysis techniques are potential areas of development. Future work could also explore integration with artificial intelligence for adaptive data embedding that dynamically adjusts to counteract potential detection methods. In this regard the exceptionally well-equipped laboratories at the Faculty of Technical Sciences at the Konstantin Preslavsky University of Shumen [5,6] give great opportunities to students majoring in "Communication and Information Systems", "Computer Technologies in Automated Manufacturing" and "Signal Security Systems and Technologies" to gain extensive theoretical and practical experience in field of the digital steganography and steganalysis [5,6].

References:

- [1] Alani, N., & Qasim, A. (2016). "Image Steganalysis and Detection of Steganographic Content: A Forensic Approach." *Digital Investigation*, 17, 1-10. ISSN: 1742-2876.
- [2] Ahmed, F., Khatr, P., Surange, G. and Agrawal, A., 2023. SearchOL: A Tool for Reconnaissance. *Journal of Network and Innovative Computing*, ISSN 2160-2174, Volume 11(2023) pp. 021-029.
- [3] Bedzhev, B., Trifonov, T., & Nikolov, N. (2010). A multicore computer system for design of stream ciphers based on random feedback shift registers. *Istanbul Aydın Üniversitesi Dergisi, Turkey*, 2(7), 1-15., <https://dergipark.org.tr/en/download/article-file/319309>. [Last accessed on 25 September 2024]
- [4] Bhuva, B. D., Zavorsky, P., & Butakov, S. (2021, May). An Analysis of Effectiveness of StegoAppDB and Data Hiding Efficiency of StegHide Image Steganography Tools. In *2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC)* (pp. 208-211). IEEE.
- [5] Boyanov, P., Using modified sniffer scripts, implementing linear algorithms for detection of network port scan attacks in Linux based operating systems. *A refereed Journal Scientific and Applied Research, Konstantin Preslavsky University Press, Vol. 24, Shumen, 2023, ISSN*

- 1314-6289 (Print), ISSN 2815-4622 (Online), pp. 78-88, DOI: <https://doi.org/10.46687/jsar.v24i1.371>.
- [6] Boyanov, P., Investigating the network traffic using the command-line packets sniffer Tcpdump in Kali Linux. A refereed Journal Scientific and Applied Research, Konstantin Preslavsky University Press, Vol. 25, Shumen, 2023, ISSN 1314-6289 (Print), ISSN 2815-4622 (Online), pp. 31-44, DOI: <https://doi.org/10.46687/jsar.v25i1.378>.
- [7] Boehm, B. (2004). Steganography in digital media: Principles, algorithms, and applications. Cambridge University Press. ISBN: 978-0521516049.
- [8] Cachin, C. (1998). An information-theoretic model for steganography. In Proceedings of the 2nd International Workshop on Information Hiding, pp. 306-318. ISBN: 978-3540656976.
- [9] Channalli, S. S., & Jadhav, A. A. (2018). "Steganography and Steganalysis: An Overview." International Journal of Computer Science and Information Security, 6(3), 244-246. ISSN: 1947-5500.
- [10] Cheddad, A., Condell, J., Curran, K., & Mc Kevitt, P. (2010). "Digital image steganography: Survey and analysis of current methods." Signal Processing, 90(3), 727-752. ISSN: 0165-1684.
- [11] Djebali, A., & Talhi, M. (2020). "Analysis of Steganographic Tools in Cybersecurity: From Embedding to Detection." In Handbook of Research on Multimedia Cyber Security, 251-267. ISBN: 978-1799842034.
- [12] Fridrich, J., Goljan, M., & Du, R. (2001). "Reliable detection of LSB steganography in color and grayscale images." In Proceedings of the 2001 Workshop on Multimedia and Security: New Challenges (pp. 27-30). ISBN: 978-1581133947.
- [13] Garg, N., & Garg, S. (2022). "A survey on image steganography techniques with focus on security and robustness." Journal of Information Security and Applications, 67, 103183. ISSN: 2214-2126.
- [14] Gueorguiev N.L., Nesterov K.N., Minev S., An approach to information exchange management in multimodule multi-position security systems. International Scientific Journal "Security & Future", Vol. 6, Issue 1, pp: 28-31, STUME, 2022, WEB ISSN 2535-082X; PRINT ISSN 2535-0668.
- [15] Gutub, A., & Al-Shaikh, A. (2019). "Security Enhancement of Data Hiding in JPEG Images: A Comparative Study of Steganography Techniques." In *Advances in Cybersecurity and Internet of Things*, 231-244. ISBN: 978-9811397364.

- [16] Iliev, R., K. Ignatova. Cloud technologies for building data center system for defense and security. T. Tagarev et al. (eds.), Digital Transformation, Cyber Security and Resilience of Modern Societies, Studies in Big Data 84, ISBN 978-3-030-65721-5, Springer 2020, pp.13-24,<https://doi.org/10.1007/978-3-030-65722-2>.
- [17] Iliev, R., Kochankov, M., A Generalized Net Model of Command and Control System. In Proceedings of the 15th International Scientific and Practical Conference, Environment. Technology. Resources. Rezekne, Latvia, Volume II, pp. 127-131, Print ISSN 1691-5402, Online ISSN 2256-070X, <https://doi.org/10.17770/etr2024vol2.8035>.
- [18] Ivanov, I., & Aleksandrova, K. (2024, June). Design and Implementation of Software-Defined Doppler Radar. In Proceedings of the 15th International Scientific and Practical Conference, Environment. Technology. Resources. Rezekne, Latvia, Volume III, pp. 105-108, Print ISSN 1691-5402, Online ISSN 2256-070X, <https://doi.org/10.17770/etr2024vol3.8159>.
- [19] Johnson, N. F., & Jajodia, S. (1998). "Exploring steganography: Seeing the unseen." Computer, 31(2), 26-34. ISSN: 0018-9162.
- [20] Kaur, G., & Kumar, A. (2018). "Comparative Analysis of Steganographic Tools Based on Least Significant Bit (LSB) Method." International Journal of Computer Science and Mobile Computing, 7(7), 134-140. ISSN: 2320-088X.
- [21] Kharrazi, M., Sencar, H. T., & Memon, N. (2004). "Image steganography: Concepts and practice." In Proceedings of the International Workshop on Digital Watermarking, pp. 42-57. ISBN: 978-3540253328.
- [22] Kochankov, M., & Iliev, R. (2024, June). A Generalized Net Model for Accessing Information Resources in a Secure Environment. In Proceedings of the 15th International Scientific and Practical Conference, Environment. Technology. Resources. Rezekne, Latvia, Volume II, pp. 175-178, Print ISSN 1691-5402, Online ISSN 2256-070X, <https://doi.org/10.17770/etr2024vol2.8034>.
- [23] Kolev, Alexander, Nikolova, Pavlina. Instrumental Equipment for Cyberattack Prevention. Information & Security: An International Journal 47, no. 3 (2020):285-299. <https://doi.org/10.11610/isij.4720>.
- [24] Komínková Oplatková, Z., Hološka, J., Zelinka, I., & Šenkeřík, R. (2009). Detection of steganography inserted by OutGuess and steghide by means of neural networks. In 2009 Third Asia International Conference on

- Modelling & Simulation, Vols 1 and 2. The Institute of Electrical and Electronics Engineers (IEEE).
- [25] Kothari, A., & Deshmukh, R. (2023). "Improving security in image steganography with advanced LSB techniques and steganalysis-resistant approaches." *International Journal of Image and Data Fusion*, 14(2), 127-141. ISSN: 1947-9895.
- [26] Kumar, M., & Reddy, B. V. (2017). "A robust and secure steganographic method using dynamic LSB technique." *International Journal of Advanced Research in Computer Science and Software Engineering*, 7(2), 79-84. ISSN: 2277-128X.
- [27] Kunčický, R., Ličev, L., & Hendrych, J. (2017). Statistical analysis of steganalytical method for Steghide Detection. In *17th International Multidisciplinary Scientific Geoconference Sgem 2017* (pp. 611-616).
- [28] Leila, Benarous., Mohamed, Djoudi., & Ahmed, Bouridane. (2016). A comparative study of steganography and steganalysis tools. *Algerian Journal of Signals and Systems*, 1(2), 92-98.
- [29] Lyu, S., & Farid, H. (2004). "Steganalysis using color wavelet statistics and one-class support vector machines." In *Proceedings of the 2004 SPIE Electronic Imaging*, pp. 35-46. ISBN: 978-0819453072.
- [30] Mirtcheva-Ivanova, D., Application of electronic platforms to increase the knowledge of learners. In *Proceedings of the 15th International Scientific and Practical Conference, Environment. Technology. Resources. Rezekne, Latvia, Volume II*, pp. 448-452, Print ISSN 1691-5402, Online ISSN 2256-070X, <https://doi.org/10.17770/etr2024vol2.8090>.
- [31] Mirtcheva-Ivanova, D., Application of Artificial Intelligence in E-Learning. In *Proceedings of the 15th International Scientific and Practical Conference, Environment. Technology. Resources. Rezekne, Latvia, Volume II*, pp. 208-211, Print ISSN 1691-5402, Online ISSN 2256-070X, <https://doi.org/10.17770/etr2024vol2.8053>.
- [32] Nishimura, A. (2019). A Novel Steganalysis of Steghide Focused on High-Frequency Region of Audio Waveform. In *Digital Forensics and Watermarking: 17th International Workshop, IWDW 2018, Jeju Island, Korea, October 22-24, 2018, Proceedings 17* (pp. 69-82). Springer International Publishing.
- [33] Oplatková, Z., Holoska, J., Zelinka, I., & Senkerik, R. (2009, May). Detection of steganography inserted by outguess and steghide by means of

- neural networks. In 2009 Third Asia International Conference on Modelling & Simulation (pp. 7-12). IEEE.
- [34] Pavlov, G., Kolev. Al., A place of GIS technologies in information Systems for crisis prevention, ICAICTSEE – 2016, December 2-3rd, 2016, UNWE, Sofia, Bulgaria, ISSN 2367-7635 (print), ISSN 2367-7643 (online), pp. 452-457.
- [35] Petitcolas, F. A. P., Anderson, R. J., & Kuhn, M. G. (1999). "Information hiding - a survey." *Proceedings of the IEEE*, 87(7), 1062-1078. ISSN: 0018-9219.
- [36] Provos, N., & Honeyman, P. (2003). "Hide and seek: An introduction to steganography." *IEEE Security & Privacy*, 1(3), 32-44. ISSN: 1540-7993.
- [37] Qureshi, M. A., Ahmed, S., Mehmood, A., Shaheen, R., & Dildar, M. S. (2024). Vulnerability assessment of operating systems in healthcare: exploitation implications techniques and security. *Health Sciences Journal*, 2(2), 104-111, ISSN (Online): 2959-2259, ISSN (Print): 2959-2240, [https://doi.org/10.59365/hsj.2\(2\).2024.98](https://doi.org/10.59365/hsj.2(2).2024.98).
- [38] Ru, X. M., Zhang, H. J., & Huang, X. (2005, August). Steganalysis of audio: Attacking the steghide. In 2005 International Conference on Machine Learning and Cybernetics (Vol. 7, pp. 3937-3942). IEEE.
- [39] Sahu, A., & Verma, A. (2019). "A novel approach for steganography in images using enhanced LSB technique." *Journal of Discrete Mathematical Sciences and Cryptography*, 22(1), 107-119. ISSN: 0972-0529.
- [40] Shaikh, S. A., & Patil, A. A. (2020). "Image Steganography: Advances and Future Research Challenges." In *Proceedings of the International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 1477-1481. ISBN: 978-1728145868.
- [41] Twain, M. (2018). Section 10.1. Accuracy and Precision. *Principles and Practice of Big Data: Preparing, Sharing, and Analyzing Complex Information*, 207.
- [42] Tzschoppe, R., Dittmann, J., Fridrich, J., & Goljan, M. (2001). "Modeling the security of steganographic systems." In *Proceedings of the SPIE International Conference on Security and Watermarking of Multimedia Contents*, pp. 16-29. ISBN: 978-0819440355.
- [43] Zhang, Z., & Xie, Y. (2021). "Steganography in Digital Images and Deep Learning-Based Detection Approaches: A Survey." *IEEE Access*, 9, 59582-59600. ISSN: 2169-3536.