Experiment 9

9.1 Aim: The objective of this experiment is to provide hands-on experience in using advanced cybersecurity tools for web application security testing and data hiding techniques. This includes:

- **Burp Suite Tool**: To conduct comprehensive web application security testing, including interception, scanning, and exploitation of vulnerabilities.
- **Steghide Tool**: To learn and apply steganographic techniques for securely hiding and extracting data within digital media files.

9.2 Course Outcomes (CO):

- **CO3:** Analyze and describe system security, malicious software, and security measures across the Network Layer, Transport Layer, and Application Layer. Focus on practical applications of these principles in real-world scenarios.
- **CO4:** Explain the importance of network management security, the role of Network Access Control (NAC), and identify the key functions of Intrusion Detection Systems (IDS) and firewalls in ensuring comprehensive system security.

9.3 Lab Objectives: This experiment aims to:

- **Web Application Security Testing with Burp Suite:** Familiarize with Burp Suite for intercepting and analyzing web traffic, performing vulnerability scans, and exploiting identified weaknesses to understand common web security issues.
- **Steganography and Data Protection:** Utilize Steghide to securely embed and extract hidden data from image and audio files, gaining insight into data concealment techniques used in cybersecurity.
- **Practical Application:** Equip students with the skills needed to perform detailed security analysis on web applications and apply methods to safeguard sensitive data through steganography.

9.4 Requirements:

- Burp Suite (available in Kali Linux)
- Steghide tool (available in Linux/Windows)
- Windows/Linux machine for execution
- Sample image/audio files for steganography
- Java (for Burp Suite)

9.5 Theory:

Burp Suite Tool: Burp Suite is a comprehensive suite of tools widely used by penetration testers, security researchers, and ethical hackers to identify vulnerabilities in web applications. Its primary function is to intercept, analyze, and manipulate HTTP/S traffic to assess the security posture of web applications.

Working Principle:

- **Proxy Functionality:** Burp Suite operates as a proxy, allowing the interception and modification of HTTP/S requests and responses between a web browser and the server.
- **Key Tools within Burp Suite:**
  - **Intercept:** Captures and modifies HTTP/S requests and responses in real-time.
  - **Scanner:** Automatically scans web applications for security vulnerabilities such as SQL injection, Cross-Site Scripting (XSS), and other weaknesses.
  - **Repeater:** Sends modified requests to test responses and analyze vulnerabilities.
  - **Intruder:** Automates attacks like brute force and fuzzing to find exploitable weaknesses.
- **Common Vulnerabilities Tested:**
  - SQL Injection
  - Cross-Site Scripting (XSS)
  - Broken Authentication
  - Cross-Site Request Forgery (CSRF)

Use Cases:

- Intercept and modify web traffic to identify potential vulnerabilities.
- Automate attack scenarios for testing application resilience against brute force and other attacks.
- Perform vulnerability scans and penetration tests to ensure web application security.

Steghide Tool: Steghide is a versatile tool for implementing steganography—the technique of hiding secret data within digital media (such as images, audio, or video files). Steghide allows for the embedding of confidential information within media files while preserving the original quality and appearance of the media.

Working Principle:

- **Data Embedding:** Steghide hides encrypted data within the media file (e.g., image or audio) by manipulating the file's least significant bits, which are not perceptible to the human eye or ear.

- **Passphrase Protection:** The hidden data is encrypted and can only be retrieved using a specific passphrase, ensuring data security and confidentiality.

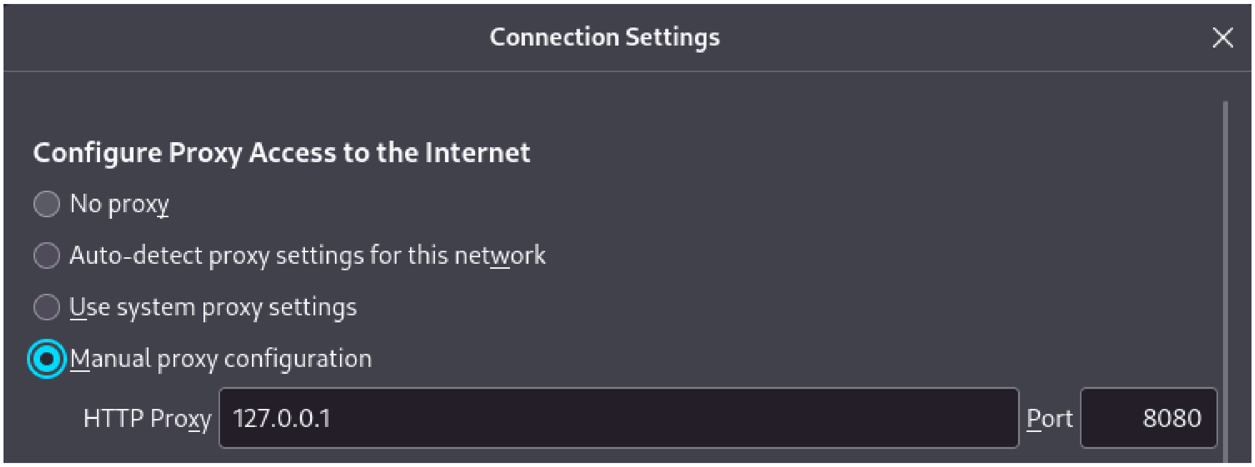
- **Supported File Formats:**
  - **Images:** JPEG, BMP, PNG
  - **Audio:** WAV, AU

Common Uses of Steghide:

- **Data Concealment for Secure Communication:** Embedding confidential messages or files within images/audio for discreet communication.
- **Forensics and Cybersecurity:** Analyzing suspicious media files to detect hidden data in digital investigations.
- **Ethical Hacking & Security Research:** Understanding and analyzing steganographic techniques for cybersecurity research and penetration testing.

9.6 Output:

Part 1: Burp Suite Web Application Security Testing



Connection Settings

Configure Proxy Access to the Internet

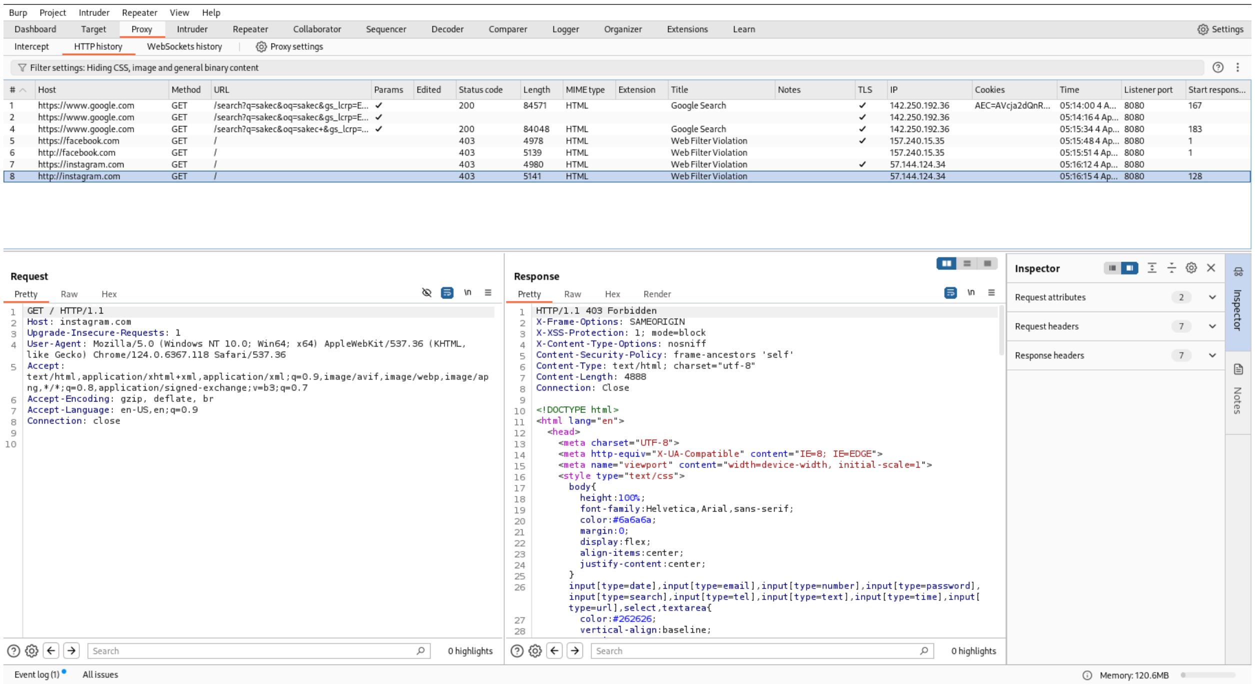
○ No proxy

○ Auto-detect proxy settings for this network

○ Use system proxy settings

● Manual proxy configuration

HTTP Proxy 127.0.0.1 Port 8080



Filter settings: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies	Time	Listener port	Start respons...
1	https://www.google.com	GET	/search?q=sakec&oq=sakec&gs_lcrp=E...	✓		200	84571	HTML		Google Search		✓	142.250.192.36	AEC=AVcja2dQnR...	05:14:00 4 A...	8080	167
2	https://www.google.com	GET	/search?q=sakec&oq=sakec&gs_lcrp=E...	✓								✓	142.250.192.36		05:14:16 4 Ap...	8080	
4	https://www.google.com	GET	/search?q=sakec&oq=sakec+&gs_lcrp=...	✓		200	84048	HTML		Google Search		✓	142.250.192.36		05:15:34 4 Ap...	8080	183
5	https://facebook.com	GET	/			403	4978	HTML		Web Filter Violation		✓	157.240.15.35		05:15:48 4 Ap...	8080	1
6	http://facebook.com	GET	/			403	5139	HTML		Web Filter Violation			157.240.15.35		05:15:51 4 Ap...	8080	1
7	https://instagram.com	GET	/			403	4980	HTML		Web Filter Violation		✓	57.144.124.34		05:16:12 4 Ap...	8080	
8	http://instagram.com	GET	/			403	5141	HTML		Web Filter Violation			57.144.124.34		05:16:15 4 Ap...	8080	128

Request

```
GET / HTTP/1.1
Host: instagram.com
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/124.0.6367.118 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close
```

Response

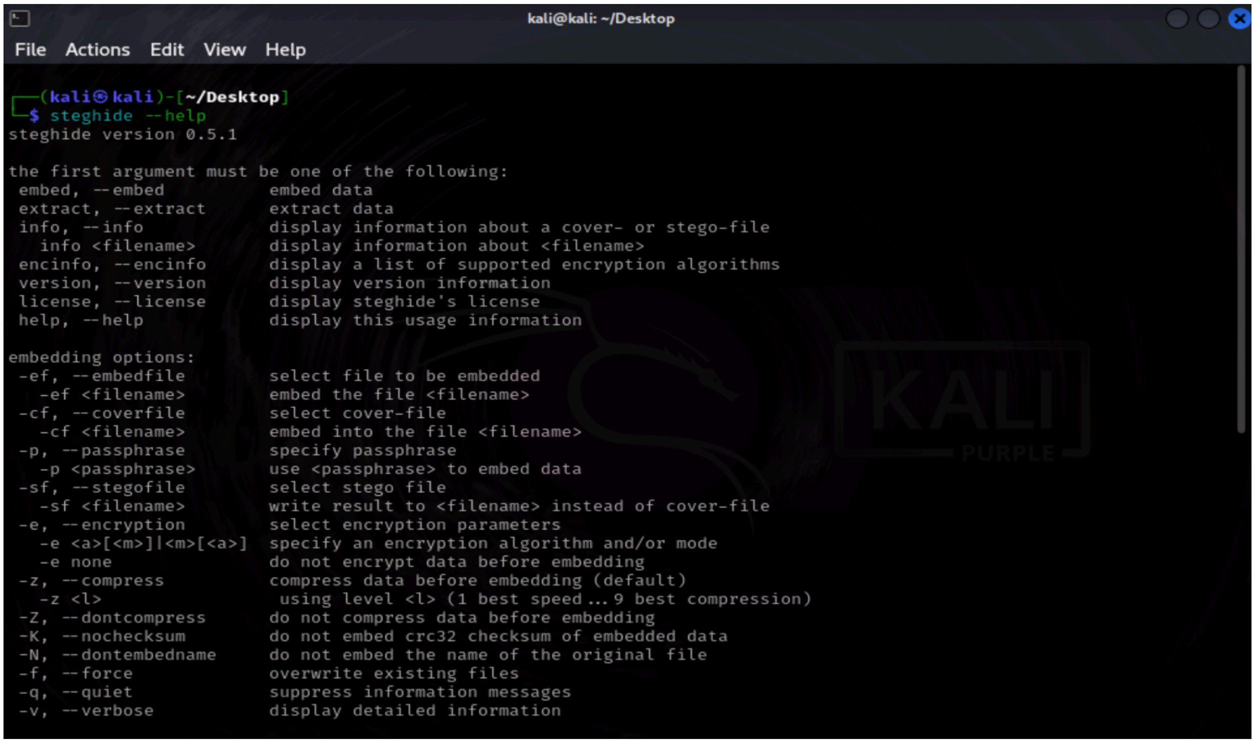
```
HTTP/1.1 403 Forbidden
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Content-Security-Policy: frame-ancestors 'self'
Content-Type: text/html; charset="utf-8"
Content-Length: 4888
Connection: Close

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=8; IE=EDGE">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <style type="text/css">
      body{
        height:100%;
        font-family:Helvetica,Arial,sans-serif;
        color:#6a6a6a;
        margin:0;
        display:flex;
        align-items:center;
        justify-content:center;
      }
      input[type=date],input[type=email],input[type=number],input[type=password],
      input[type=search],input[type=tel],input[type=text],input[type=time],input[
      type=url],select,textarea{
        color:#262626;
        vertical-align:baseline;
```

Inspector

Request attributes 2
Request headers 7
Response headers 7

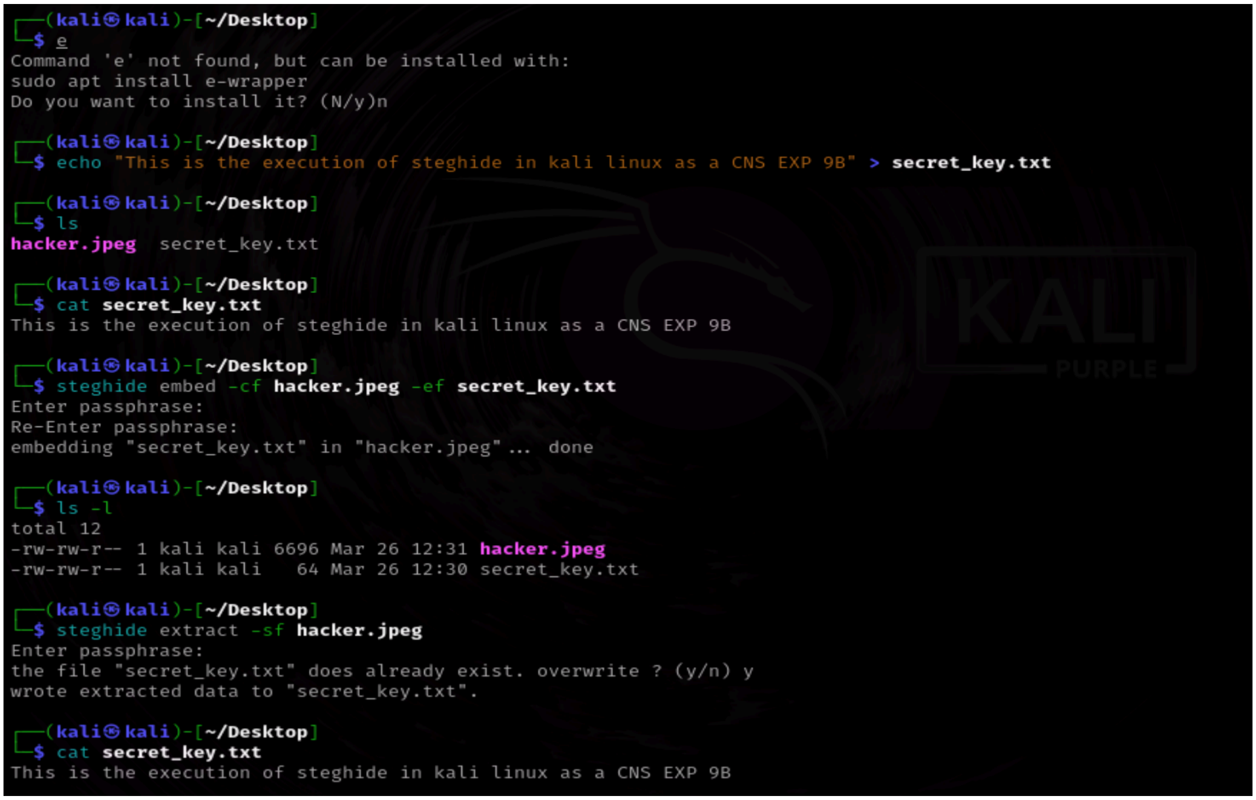
Part 2: Steghide – Data Hiding in Media Files



```
┌──(kali㉿kali)-[~/Desktop]
└─$ 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
```



```
┌──(kali㉿kali)-[~/Desktop]
└─$ e
Command 'e' not found, but can be installed with:
sudo apt install e-wrapper
Do you want to install it? (N/y)n

┌──(kali㉿kali)-[~/Desktop]
└─$ echo "This is the execution of steghide in kali linux as a CNS EXP 9B" > secret_key.txt

┌──(kali㉿kali)-[~/Desktop]
└─$ ls
hacker.jpeg  secret_key.txt

┌──(kali㉿kali)-[~/Desktop]
└─$ cat secret_key.txt
This is the execution of steghide in kali linux as a CNS EXP 9B

┌──(kali㉿kali)-[~/Desktop]
└─$ steghide embed -cf hacker.jpeg -ef secret_key.txt
Enter passphrase:
Re-Enter passphrase:
embedding "secret_key.txt" in "hacker.jpeg"... done

┌──(kali㉿kali)-[~/Desktop]
└─$ ls -l
total 12
-rw-rw-r-- 1 kali kali 6696 Mar 26 12:31 hacker.jpeg
-rw-rw-r-- 1 kali kali   64 Mar 26 12:30 secret_key.txt

┌──(kali㉿kali)-[~/Desktop]
└─$ steghide extract -sf hacker.jpeg
Enter passphrase:
the file "secret_key.txt" does already exist. overwrite ? (y/n) y
wrote extracted data to "secret_key.txt".

┌──(kali㉿kali)-[~/Desktop]
└─$ cat secret_key.txt
This is the execution of steghide in kali linux as a CNS EXP 9B
```

9.7 Conclusion:

In this experiment, two essential cybersecurity tools were performed: Burp Suite and Steghide. Burp Suite enabled the analysis of web application security and the identification of vulnerabilities, enhancing the understanding of detecting and mitigating risks such as SQL injection, XSS, and other common threats. Meanwhile, Steghide introduced steganographic techniques, allowing the secure hiding and extraction of data within digital media files. By utilizing these tools, practical insights into web application security testing and data protection methods were gained, improving knowledge of ethical hacking, penetration testing, and information concealment.