Advance Linux Exploits

Here, we discuss the following topics:

• Format string exploits

• Memory protection schemes

Format string exploits

Format String Exploits Format string exploits became public in late 2000. Unlike buffer overflows, format string errors are relatively easy to spot in source code and binary analysis. In spite of this, they are still common in applications today. Many organizations still don't utilize code analysis or binary analysis tools on software before releasing it, so these errors still occur in the wild. Once spotted, they are usually eradicated quickly. As more organizations start to use code analysis tools as part of their build processes, these types of attacks will continue to decline. However, this attack vector is still fairly easy to find and can result in some interesting code execution.

Format Strings Format strings are used by various print functions, and these functions may behave in many ways depending on the format strings provided.

- printf() Prints output to the standard input/output (STDIO) handle (usually the screen)
    - fprintf() Prints output to a file stream
    - sprintf() Prints output to a string
    - snprintf() Prints output to a string with length checking built in

When someone calls one of these functions, the format string dictates how and where the data is compiled into the final string. Format strings are very versatile, though, and if the creator of the application allows data specified by the end user to be used directly in one of these format strings, the user can change the behavior of the application. This can include disclosing additional information that the creator did not want disclosed, such as memory locations, data variables, and stack memory.

The Problem

```
printf(<format string>, <list of variables/values>);
printf(<user supplied string>);
```

the printf() function can have any number of arguments. We will discuss two forms here: In the first example, the programmer has specified a format string and then the variables that will fill in the spaces designated by the format string for data. This prevents unexpected behavior from printf. The second example allows the user to specify the format string. This means that a user can cause the printf function to behave however they want.

Format Symbol	Meaning	Example
\n	Carriage return/newline	**printf("test\n");** Result: The application prints **test**.
%d	Decimal value	**printf("test %d", 123);** Result: The application prints **test 123.**
%s	String value	**printf("test %s", "123");** Result: The application prints **test 123.**
%x	Hex value	**printf("test %x", 0x123);** Result: The application prints **test 123.**
%hn	Print the length of the current string in bytes to **var** (short int value, overwrites 16 bits)	**printf("test %hn", var);** Result: The value **04** is stored in **var** (that is, 2 bytes).
%<number>$	Direct parameter access	**printf("test %2$s", "12", "123");** Result: **test 123** (the second parameter is used directly and then treated as a string).

The Correct Way

Recall the correct way to use the printf() function.

For example, the code

```
//fmt1.c
#include <stdio.h>
int main() {
  printf("This is a %s.\n", "test");
  return 0;
}
```

produces the following output:

```
#gcc -o fmt1 fmt1.c
#./fmt1
This is a test.
```

The Incorrect Way

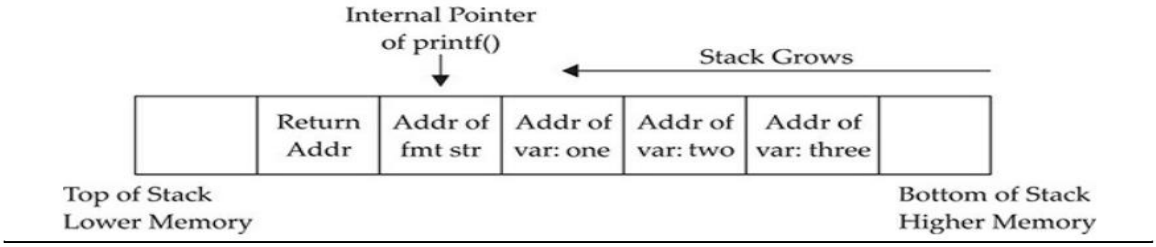
Now take a look at what happens if we forget to add a value for %s to replace:

```
// fmt2.c
#include <stdio.h>
int main() {
   printf("This is a %s.\n");
   return 0;
}
```

```
# gcc -o fmt2 fmt2.c
#./fmt2
This is a ey¿.
```

```
//fmt4.c
#include <stdio.h>
int main(){
       int one=1, two=2, three=3;
       printf("Testing %d, %d, %d!\n", one, two, three);
       return 0;
}
$gcc -o fmt4.c
./fmt4
Testing 1, 2, 3!
```

During execution of the **printf()** function, the stack looks like Figure 12-1. As always, the parameters of the **printf()** function are pushed on the stack in reverse order, as shown in the figure. The addresses of the parameter variables are used. The **printf()** function maintains an internal pointer that starts out pointing to the format string (or top of the stack frame) and then begins to print characters of the format string to the STDIO handle (the screen in this case) until it comes upon a special character.



printf() function maintains an internal pointer that starts out pointing to the format string (or top of the stack frame) and then begins to print characters of the format string to the STDIO handle (the screen in this case) until it comes upon a special character. If the % is encountered, the printf() function expects a format token to follow and thus increments an internal pointer (toward the bottom of the stack frame) to grab input for the format token (either a variable or absolute value).

Therein lies the problem: the printf() function has no way of knowing if the correct number of variables or values were placed on the stack for it to operate. If the programmer is sloppy and does not supply the correct number of arguments, or if the user is allowed to present their own format string, the function will happily move down the stack (higher in memory), grabbing the next value to satisfy the format string requirements. So what we saw in our previous examples was the printf() function grabbing the next value on the stack and returning it where the format token required.

Reading from Arbitrary Memory

We will now begin to take advantage of the vulnerable program. We will start slowly and then pick up speed. Using the %x Token to Map Out the Stack As shown in previous Table, the %x format token is used to provide a hex value. So, by supplying a few %08x tokens to our vulnerable program, we should be able to dump the stack values to the screen:

$ ./fmtstr "AAAA %08x %08x %08x %08x"

AAAA bffffd2d 00000648 00000774 41414141

Canary at 0x08049440 = 0x00000000

$

The 08 is used to define precision of the hex value (in this case, 8 bytes wide). Notice that the format string itself was stored on the stack, proven by the presence of our AAAA (0x41414141) test string. The fact that the fourth item shown (from the stack) was our format string depends on the nature of the format function used and the location of the vulnerable call in the vulnerable program. To find this value, simply use brute force and keep increasing the number of %08x tokens until the beginning of the format string is found. For our simple example (fmtstr), the distance, called the offset, is defined as 4.

Using the %s Token to Read Arbitrary Strings

Because we control the format string, we can place anything in it we like (well, almost anything). For example, if we wanted to read the value of the address located in the fourth parameter, we could simply replace the fourth format token with a %s, as shown:

$ ./fmtstr "AAAA %08x %08x %08x %s"

Because, as you recall, the %s format token will take the next parameter on the stack, in this case the fourth one, and treat it like a memory address to read from (by reference). In our case, the fourth value is AAAA, which is translated in hex to 0x41414141.

Social Engineering in Kali Linux

The term **"social engineering"** is derived from the words **"social"** and **"engineering,"** where **"social"** refers to personal, professions, and our day-in-day-out lives. On the other hand, **"engineering"** involves comprehensive processes to complete a work such that the defined goal is met. In other words, it is a set of methods.

When social and engineering is combined, we get social engineering, which involves intrusion based on human interaction. It is a non-technical intrusion in which a person is often tricked into breaking the general security guidelines already set in an institution.

Social Engineering Toolkit

Social engineering toolkit is a **free and open-source tool** which is used for social engineering attacks like **phishing, sending SMS, faking phone,** etc. It is a free tool that comes with Kali Linux, or we can download and install it directly from **Github.** The Social Engineering Toolkit is designed and developed by a programmer named **Dave Kennedy.** Security researchers and penetration testers use this tool to check cybersecurity issues in systems all over the world. The goal of the social engineering toolkit is to perform attacking techniques on their machines. This toolkit also includes website vector attacks and custom vector attacks, which allow us to **clone any website, perform phishing attacks.**

Features of Social Engineering Toolkit

The following are the features of the social engineering toolkit:

- Social Engineering Toolkit is **free** and **open source.**
- Social Engineering Toolkit is portable, which means we can quickly switch attack vectors.
- Social Engineering Toolkit supports integration with third-party modules.
- Social Engineering Toolkit is already installed in our Kali Linux, but we can also download and install it from **Github.**
- Social Engineering Toolkit is a **multi-platform** tool; we can run it in **Windows, Linux,** and **Unix.**
- Social Engineering Toolkit contains access to the **Fast-Track Penetration Testing platform.**
- Social Engineering Toolkit offers various attack vectors like **Website Attacks, Infection Media Generator, Website Attacks,** etc.

Uses of Social Engineering Toolkit

There are various uses of social engineering toolkit:

1. Web Attack
2. Mass Mailer Attack
3. Phishing Attacks
4. Create a Payload and Listener

1. Web Attack

In SET, a web attack is a module. This module combines various options to attack the victim remotely. Using this module, we can create a payload and distribute the payload to our victim browser using the **Metasploit browser exploit.** Web attack has **Credential Harvester method** that allows us to clone any website for a phishing attack and send the link of that webpage to the victim to get information from user and password fields.

2. Phishing Attacks

We can use the Social Engineering Toolkit to perform phishing attacks on our victims. Using SET, we can create phishing pages for a variety of websites, including **Google, Facebook, Instagram,** etc. SET will generate a link of the option which we have selected, and then we can send that URL to the victim once the victim clicks on that URL and he/she will see a

legitimate webpage of a real website that is essentially a phishing page. Once he/she has entered his/her ID password, we will get that ID password on our terminal screen, this is how a phishing attack using SET works.

3. Create a Payload and Listener

When we execute the Social Engineering Toolkit for the first time. We will see **option 4th** which is used to generate a payload and listener by using that SET module, we may develop malicious payloads for Windows, including **Shell Reverse_TCP, Shell Reverse_ HTTPS TCP X64, Meterpreter Reverse,** and **Reverse_TCP Meterpreter.** These payloads can be used in the same way that we use metasploitable payloads.

4. Mass Mailer Attack

In the social engineering toolkit, mass mailer attacker is a module which we used to send a large number of emails on target mail account, for which we can also use our own Gmail account or we can own a server for that.

These are some of the attack vectors which we can use using the Social Engineering Toolkit. When we will run the SET, we will enjoy it because it is quite simple to use.

Phases in Social Engineering

There are various phases of social engineering before the final result is obtained. This includes:

1. Research Phase
2. Hook Phase
3. Play Phase
4. Exit Phase

1. Research Phase

In the research phase, the information related to the goal is collected. Whether the objective is a firm or an individual, the first phase is the same. There are so many ways by which attackers can get the information related to their targets. These include obtaining documents from the public domain, visiting the website for the institution concerned, and in some cases, constructive face-to-face interactions. Besides, dumpster diving is also necessary at this stage of the attack.

2. Hook Phase

The Hook phase is the second phase of the attack. In this phase, the attacker initiates a discussion with their victim target.

After the hook, the phase is the phase of play that strengthens the connection between the attacker and the target. The attacker takes advantage of this opportunity to investigate getting the information they desire.

3. Exit Phase

This is the final phase, and the attacker must be careful not to set up a situation that would make the target suspect in any manner. The idea is to exit the target without giving any indication of action.

We can start these steps through various social engineering tools which are pre-installed in Kali Linux, while other tools need to be installed manually.

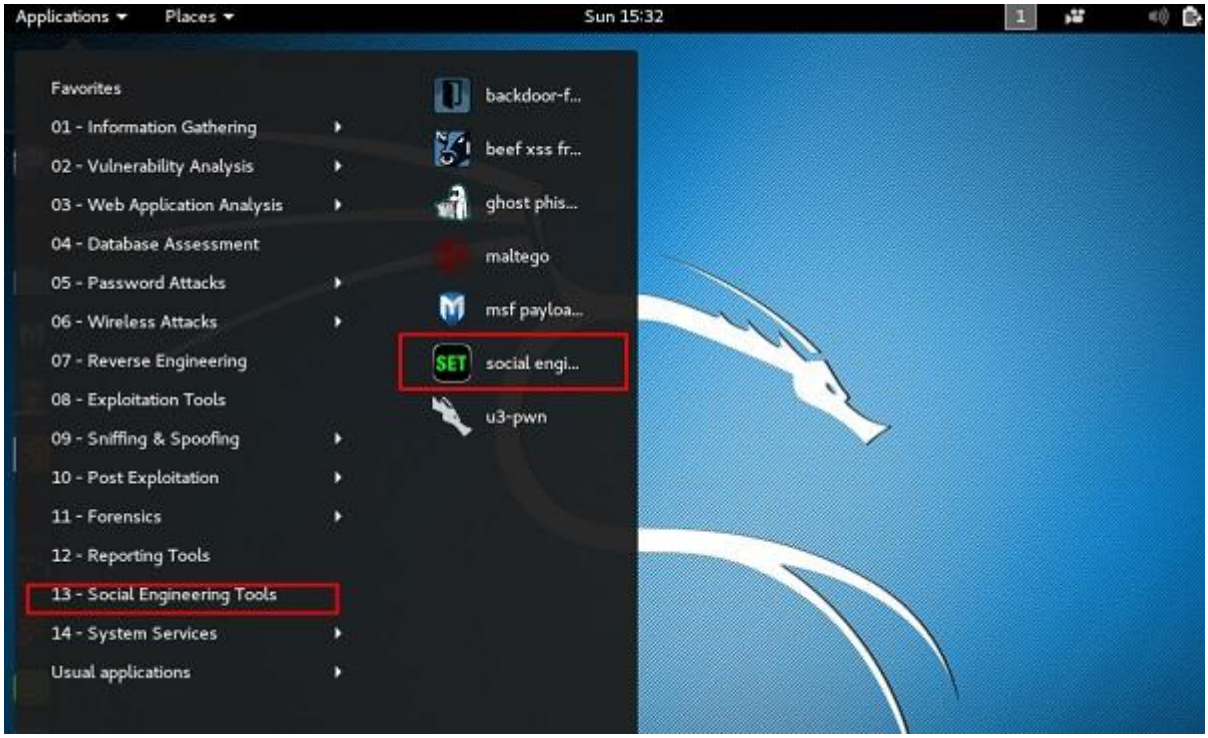
We don't have to be concerned because Social Engineering Toolkit is an open-source penetration testing platform that focuses on social engineering. This suite of tools has been fine-tuned to allow us to launch attacks in a matter of seconds.

Social Engineering Toolkit Usage

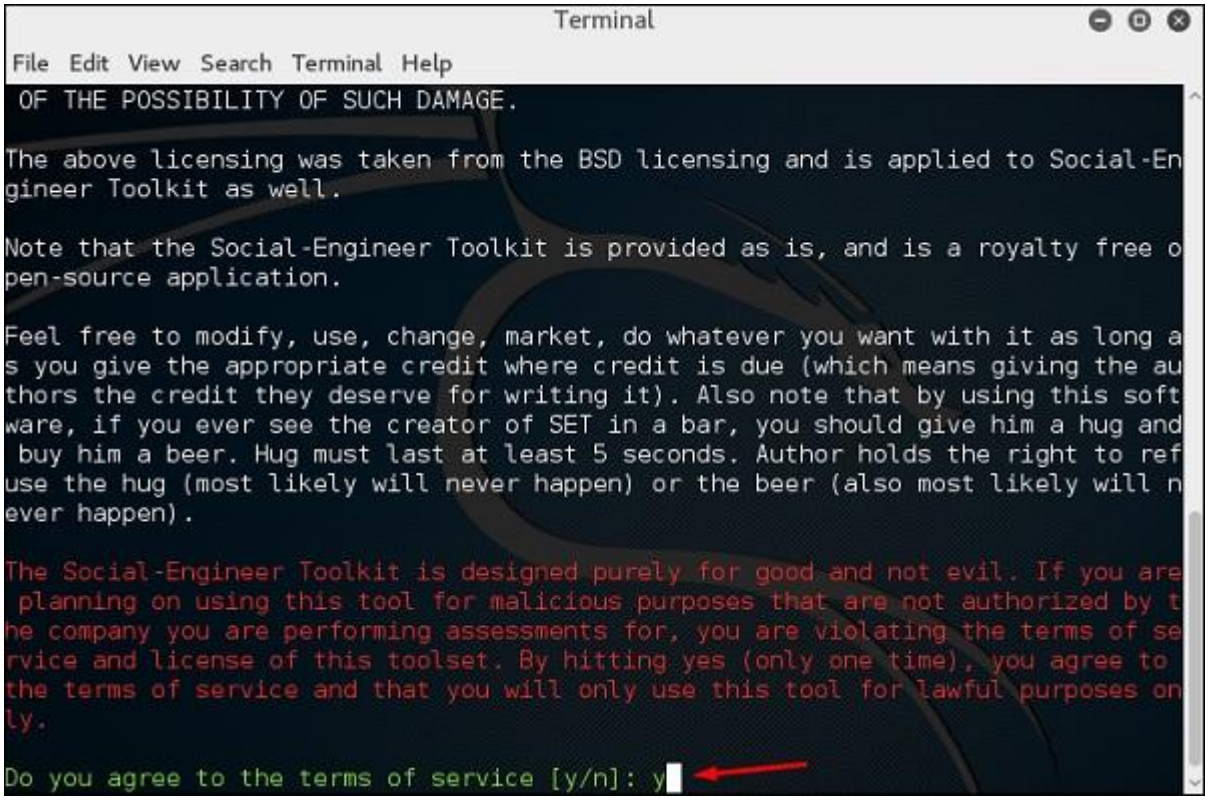
The **Social-Engineer Toolkit** (SET) is an open-source penetration testing framework designed for social engineering. SET has a number of custom attack vectors that allow you to make a believable attack in a fraction of time. These kind of tools use human behaviors to trick them to the attack vectors.

Let’s learn how to use the Social Engineer Toolkit.

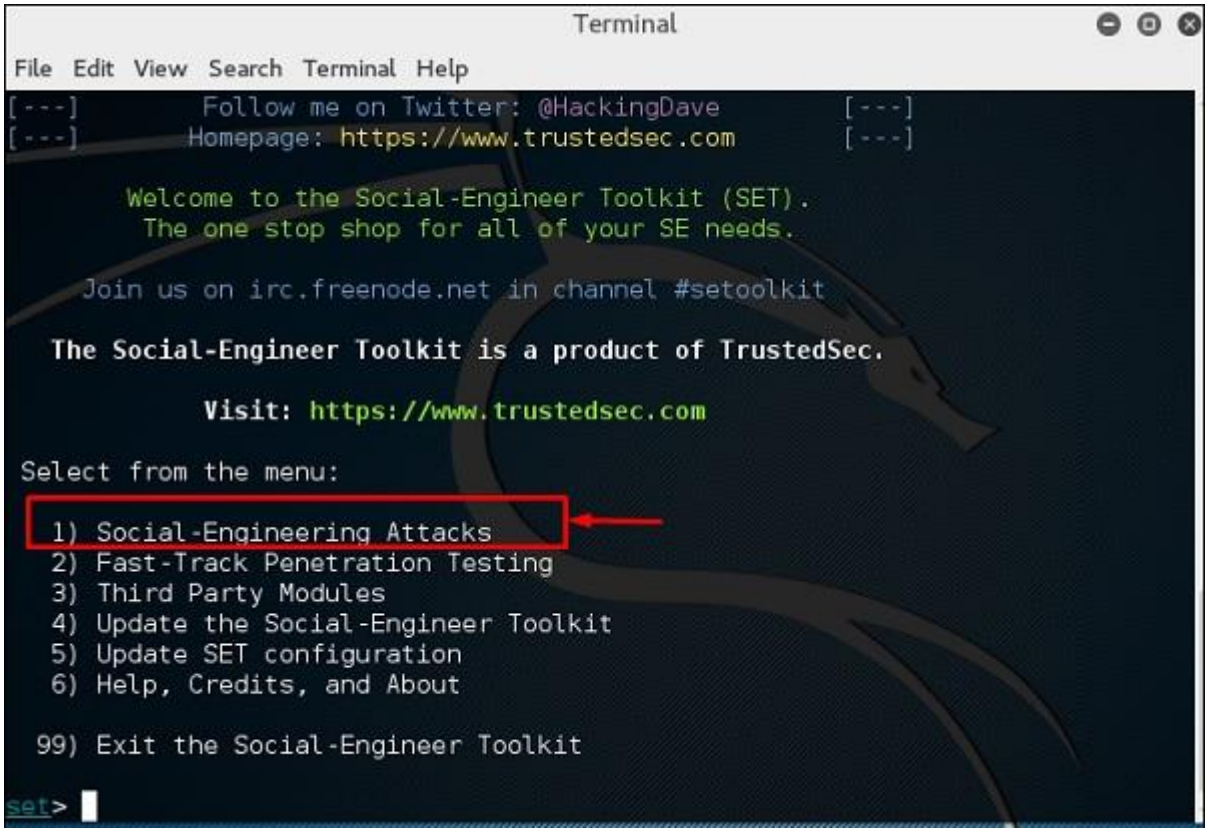
Step 1 − To open SET, go to Applications → Social Engineering Tools → Click “SET” Social Engineering Tool.



Step 2 − It will ask if you agree with the terms of usage. Type **"y"** as shown in the following screenshot.



Step 3 − Most of the menus shown in the following screenshot are self-explained and among them the most important is the number 1 "Social Engineering Attacks".



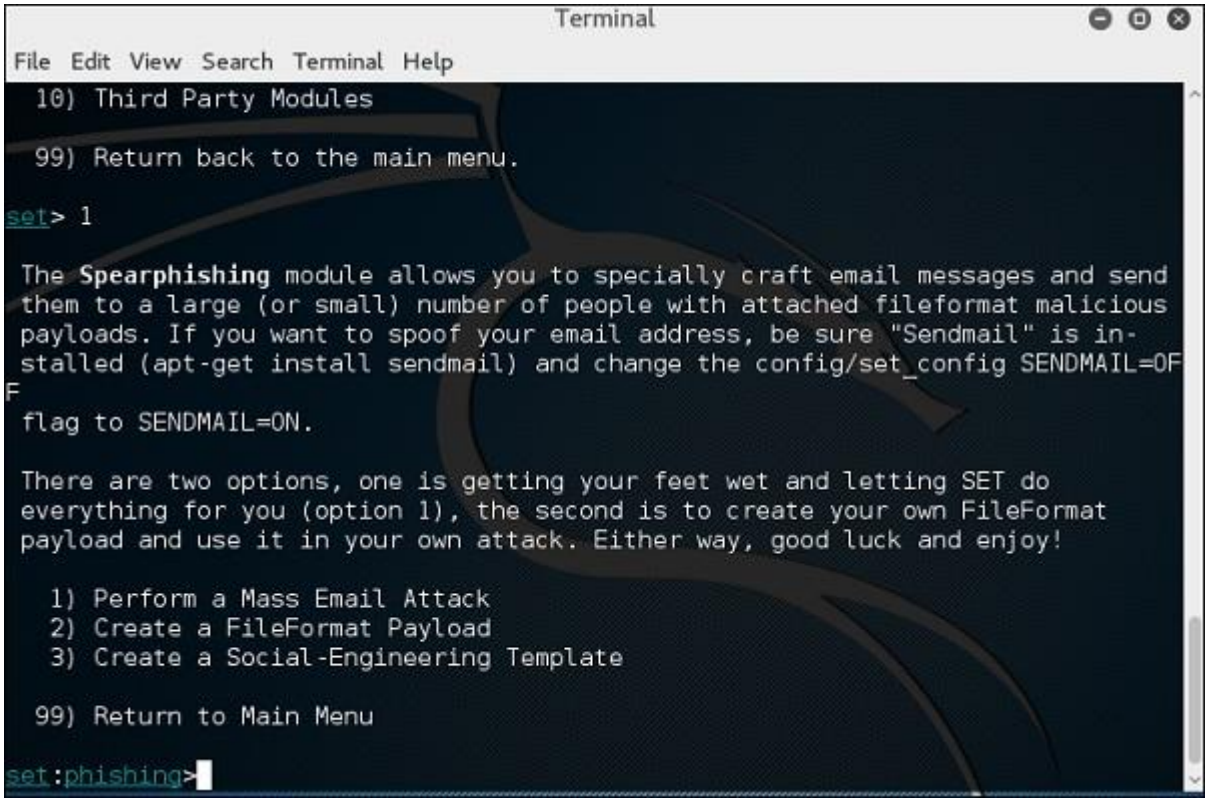
Step 4 − Type **"1"** → Enter. A submenu will open. If you press the **Enter** button again, you will see the explanations for each submenu.

The Spear-phishing module allows you to specially craft email messages and send them to your targeted victims with attached **FileFormatmalicious** payloads. For example, sending malicious PDF document which if the victim opens, it will compromise the system. If you want to spoof your email address, be sure "Sendmail" is installed (apt-get install sendmail) and change the config/set_config SENDMAIL=OFF flag to SENDMAIL=ON.

There are two options for the spear phishing attack −

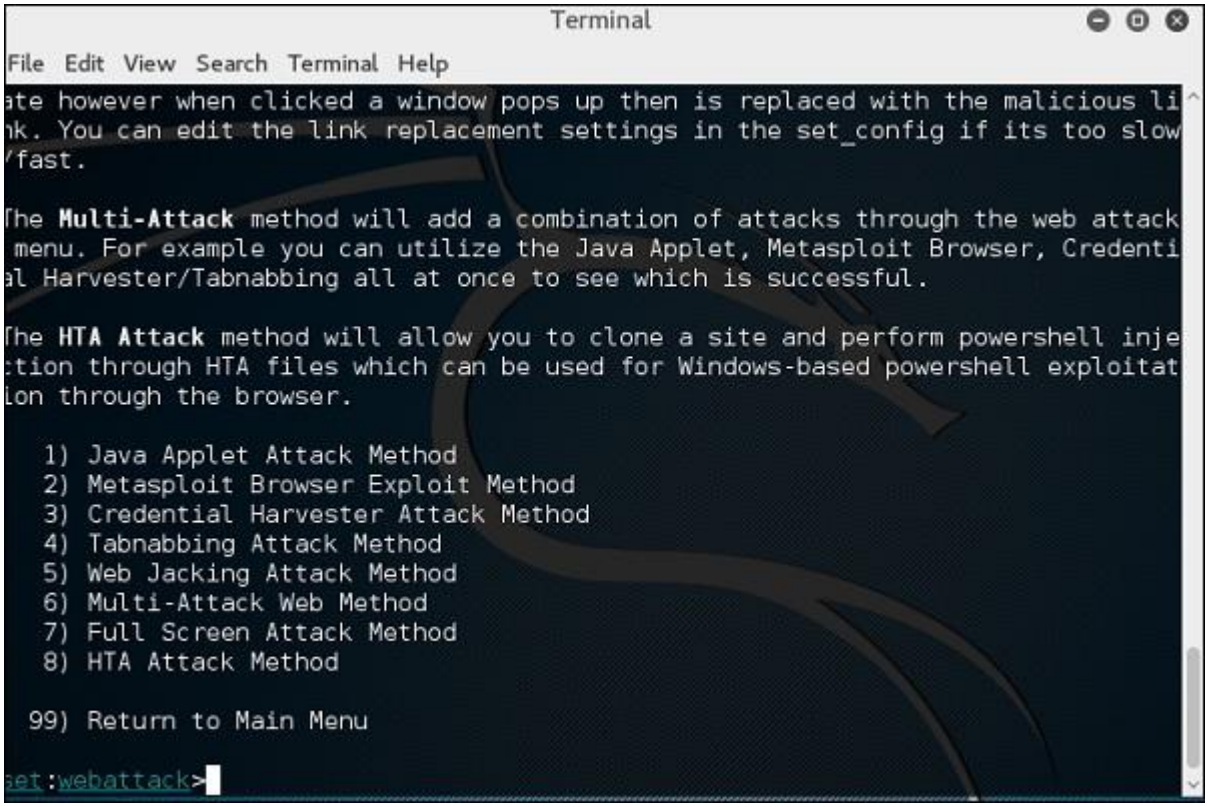
- Perform a Mass Email Attack
- Create a FileFormat Payload and a Social-Engineering Template

The first one is letting SET do everything for you (option 1), the second one is to create your own FileFormat payload and use it in your own attack.



Type **“99”** to go back to the main menu and then type **“2”** to go to “The web attack vectors”.

The web attack module is a unique way of utilizing multiple web-based attacks in order to compromise the intended victim. This module is used by performing phishing attacks against the victim if they click the link. There is a wide variety of attacks that can occur once they click a link.

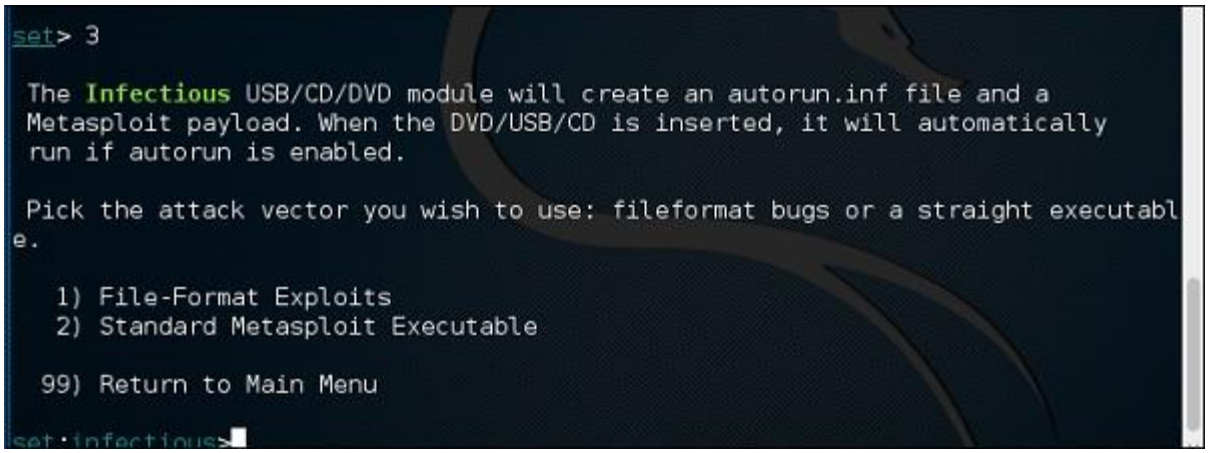


Type **"99"** to return to the main menu and then type **"3"**.

The infectious USB/CD/DVD module will create an autorun.inf file and a Metasploit payload. The payload and autorun file is burned or copied on a USB. When DVD/USB/CD is inserted in the victim's machine, it will trigger an autorun feature (if autorun is enabled) and hopefully compromise the system. You can pick the attack vector you wish to use: fileformat bugs or a straight executable.

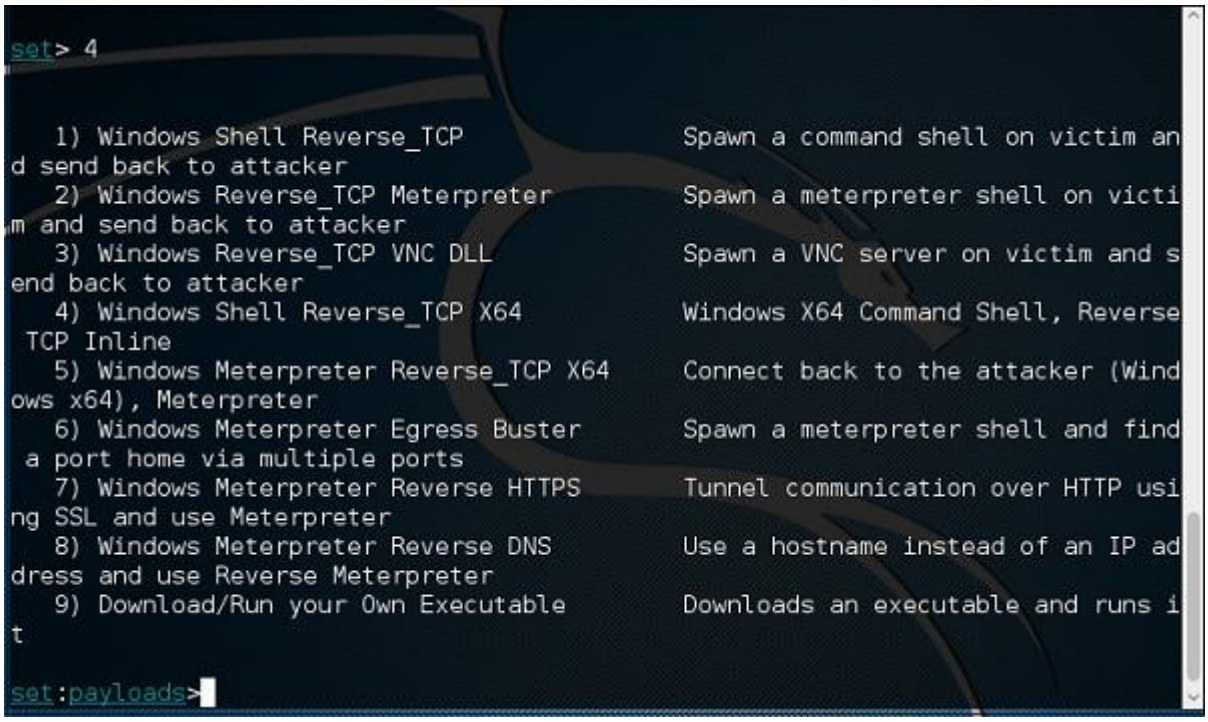
Following are the options for Infectious Media Generator.

- File-Format Exploits
- Standard Metasploit Executable

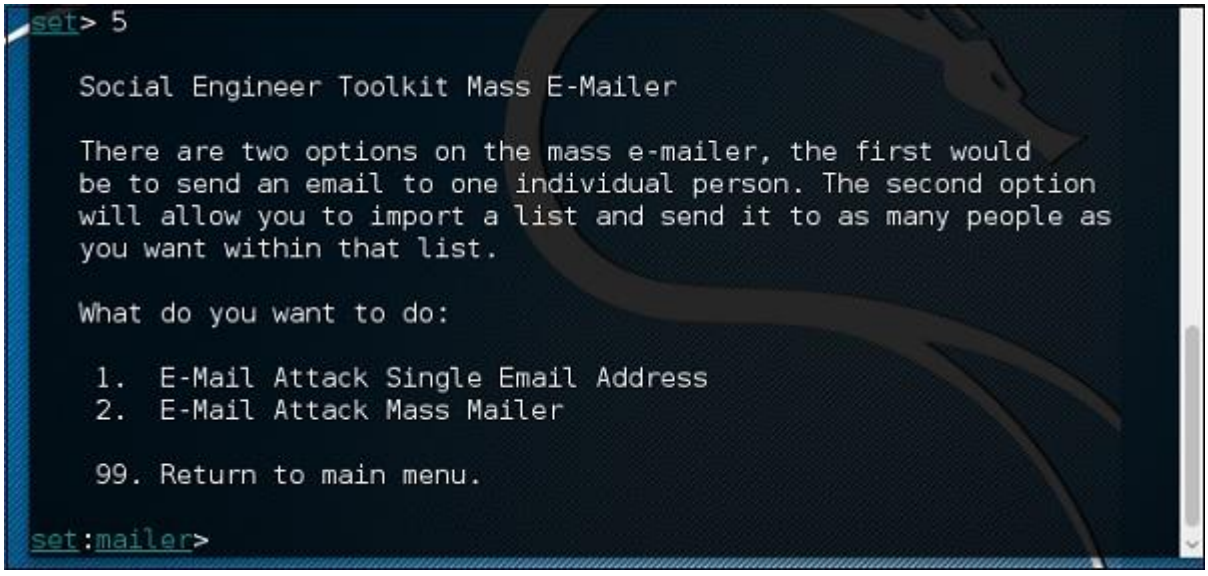


Type **"99"** to go back to the main menu. Then, type **"4"** to go to "The web attack vectors".

The create payload and listener is a simple way to create a Metasploit payload. It will export the exe file for you and generate a listener. You would need to convince the victim to download the exe file and execute it to get the shell.



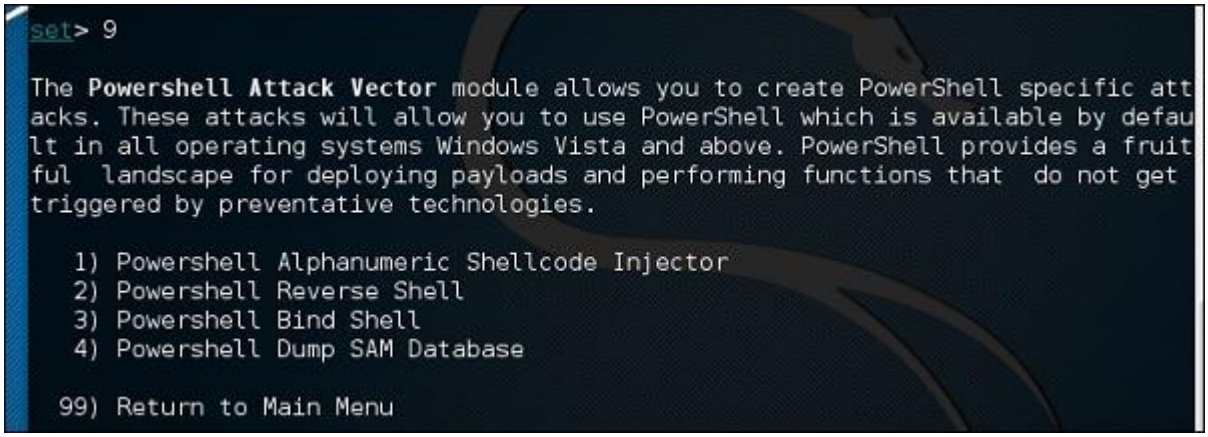
Type **"99"** to go back to the main menu and then type **"5"** to go to "The web attack vectors".



The mass mailer attack will allow you to send multiple emails to victims and customize the messages. There are two options on the mass e-mailer; the first is to send an email to a single email address. The second option allows you to import a list that has all recipient emails and it will send your message to as many people as you want within that list.

- E-Mail Attack Single Email Address
- E-Mail Attack Mass Mailer

Type **"99"** to go back to the main menu and then type **"9"** to go to "Powershell Attack Vector".



The Powershell Attack Vector module allows you to create PowerShell specific attacks. These attacks allow you to use PowerShell, which is available by default in all operating systems Windows Vista and above. PowerShell provides a fruitful landscape for deploying payloads and performing functions that do not get triggered by preventive technologies.

- Powershell Alphanumeric Shellcode Injector
- Powershell Reverse Shell
- Powershell Bind Shell
- Powershell Dump SAM Database