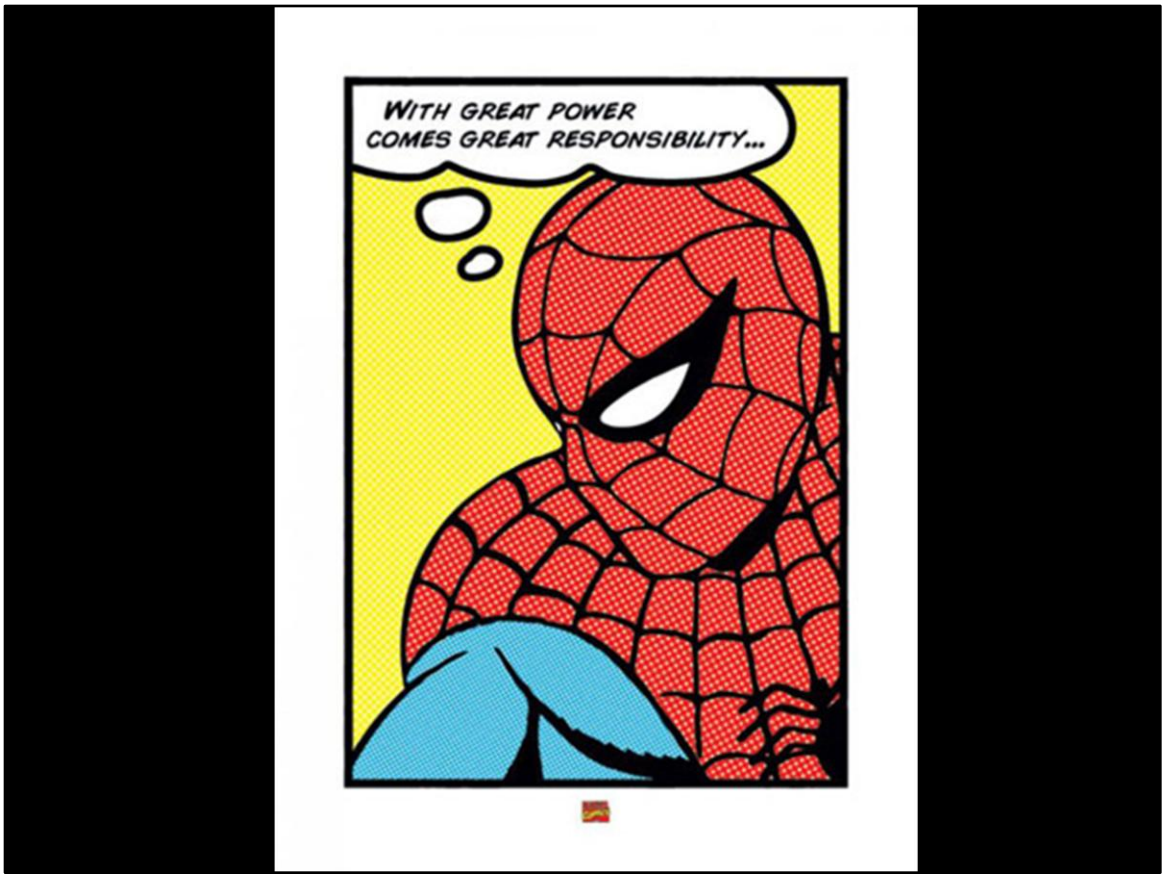
-First of a two part series
-Tonight we will look at how an attacker can gain access to your Wifi network
-Another night (maybe next month) we will look at more advanced Wifi hacks, that are more targeted against other Wifi clients – not the network itself (so all the thefts of passwords, MITM attacks will be in Wifi Attacks 102 – hey I only have an hour ☺)



Insert Standard Ethical Hacking Disclaimer here

So lets start off by looking at what's in my personal Wifi attacking kit. Here you can see a tablet, some antenna and a bunch of cables.

But don’t worry you won’t be needing all of that stuff, I’ll get to that in a minute. But first lets talk about the tablet itself ....

The tablet is the 1st generation Google Nexus 7 32GB 3G version, which is made by Asus.

The 3G here is quite useful as it means you don't need to rely on Wifi for connecting to the web AND you can connect to the device over SSH and the mobile network (e.g. if you leave your Nexus 7 hidden connected to a network on a pen-test).

The Tablet has been reflashed from the original to have a custom Firmware. Essentially its still an Android frontend, but it also has an Ubuntu backend in a CHROOT environment. So you have the benefit of having both linux and android. Most of the tools on the device use the linux part, but there are some useful Android related tools as well. And of course you can use the tablet for all the normal things you can use an Android tablet for.

You can pickup one of these on eBay for under €150 and the custom firmware is free.

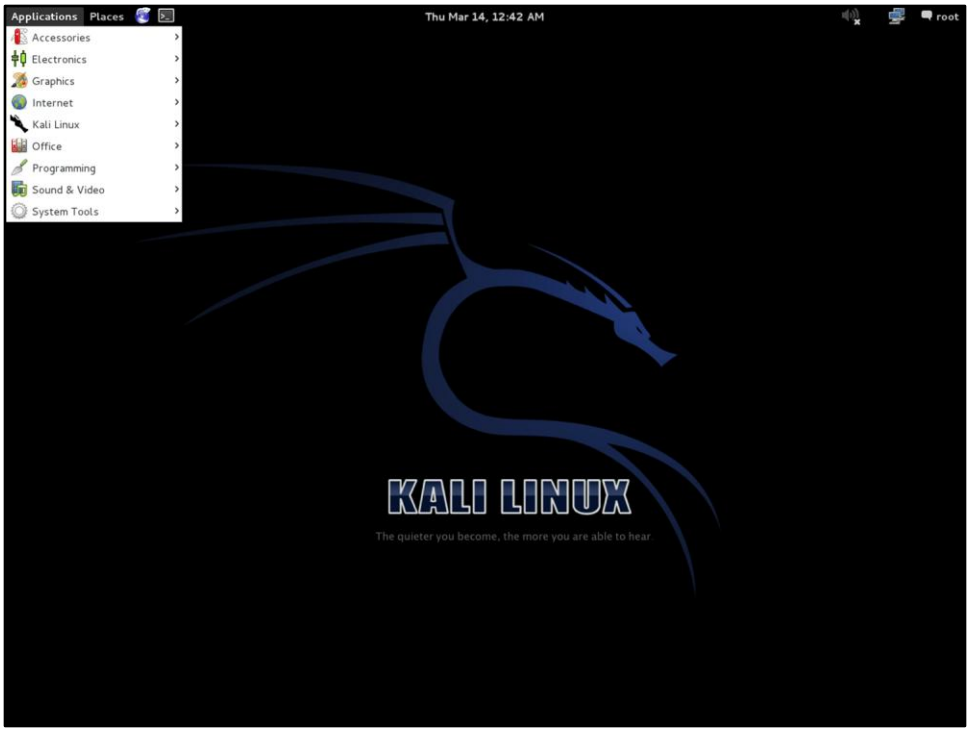
Next lets take a look at the various Adapters and Cables I have

Up the top you can see some plugs and a very handy TP-Link NANO Router, which is great for testing attacks.

Apart from this there is a **Sena UD100 Bluetooth** for Bluetooth testing (the red one) and a couple of other cables.

The only things you will be needing to get however is the one highlighted in red.

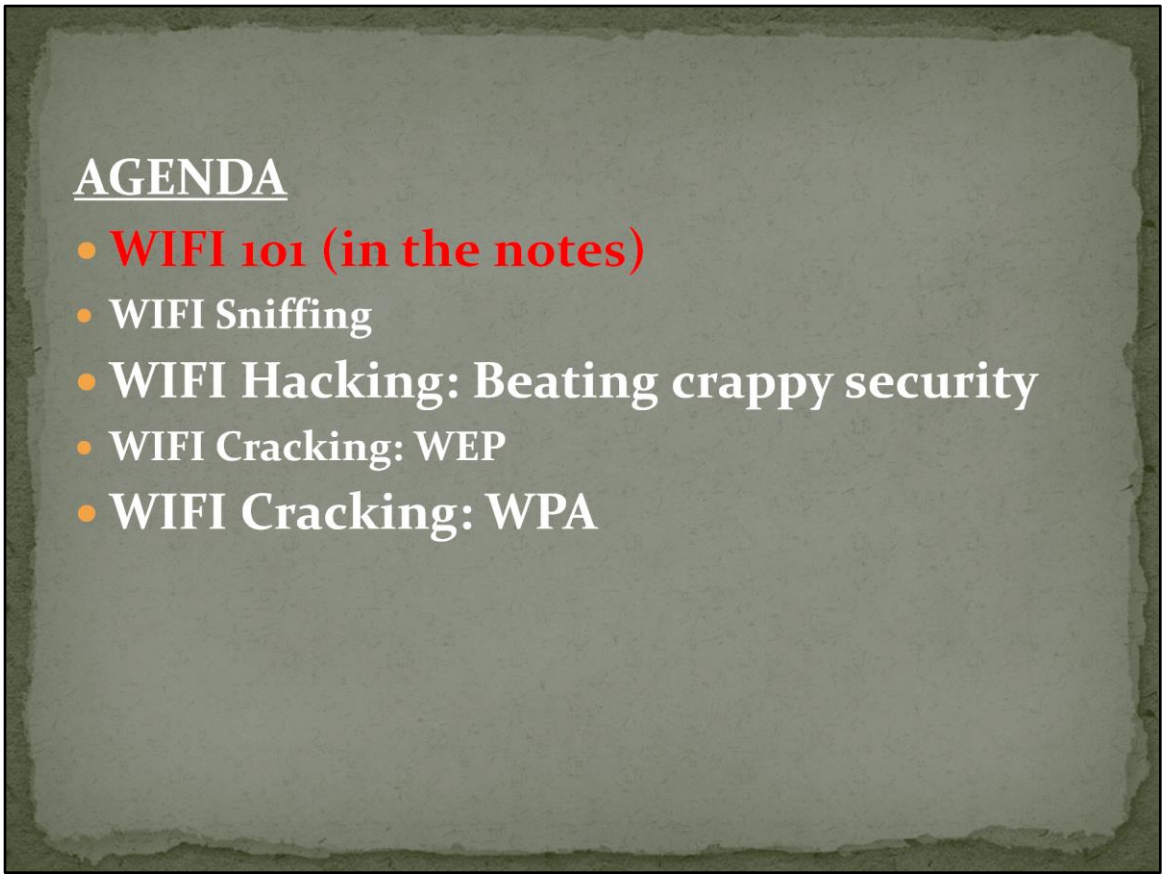
TP-Link TL-WN722N USB adapter
Has Tplink 80211n Atheros Chipset. This wifi card supports injecting packets, promiscous mode and monitor mode – which we want for our testing. You can also add other antenna – e.g. a directional antenna or higher gain antenna. The standard antenna is 4dBi. It costs around €15. There are other Wifi cards that also are great for wifi attacks, such as those made by Alfa – but this is my current personal favourite.



You do not even need the tablet! Simply using the USB adapter with a Kali Linux machine will also do the job nicely.

For the purposes of tonight though, I'm going to use the tablet because I find it handier. Also I've found that using this wireless adapter with a VM of Kali, as opposed to a full install causes issues. But be aware that everything I'm showing you using the tablet is just as easy to Kali using the same adapter.

Once loaded with the PWNPAD firmware the tablet also comes with a bunch of other awesome software such as Metasploit, Ettercap and others – but I'm not going to talk about any of that tonight



So what will we cover today?

Because I only have an hour I'm going to gloss over some of the basics of working with networking in linux. Here I'm talking about commands like ifconfig, iwconfig , how to bring up a network card, how to assign it an IP etc. I'm assuming a lot of people here have used linux a bit in the past. For those not that comfortable with linux though I'll be including slides with step by step on this stuff when I upload the notes for this talk anyhow, so don't worry.

So we'll start very simple – by looking at sniffing under Wireless to capture packets out of thin air

We will then look as some crappy wifi security techniques, and how to by pass them

And then the rest of today will focus on breaking WEP and WPA to gain access to a network

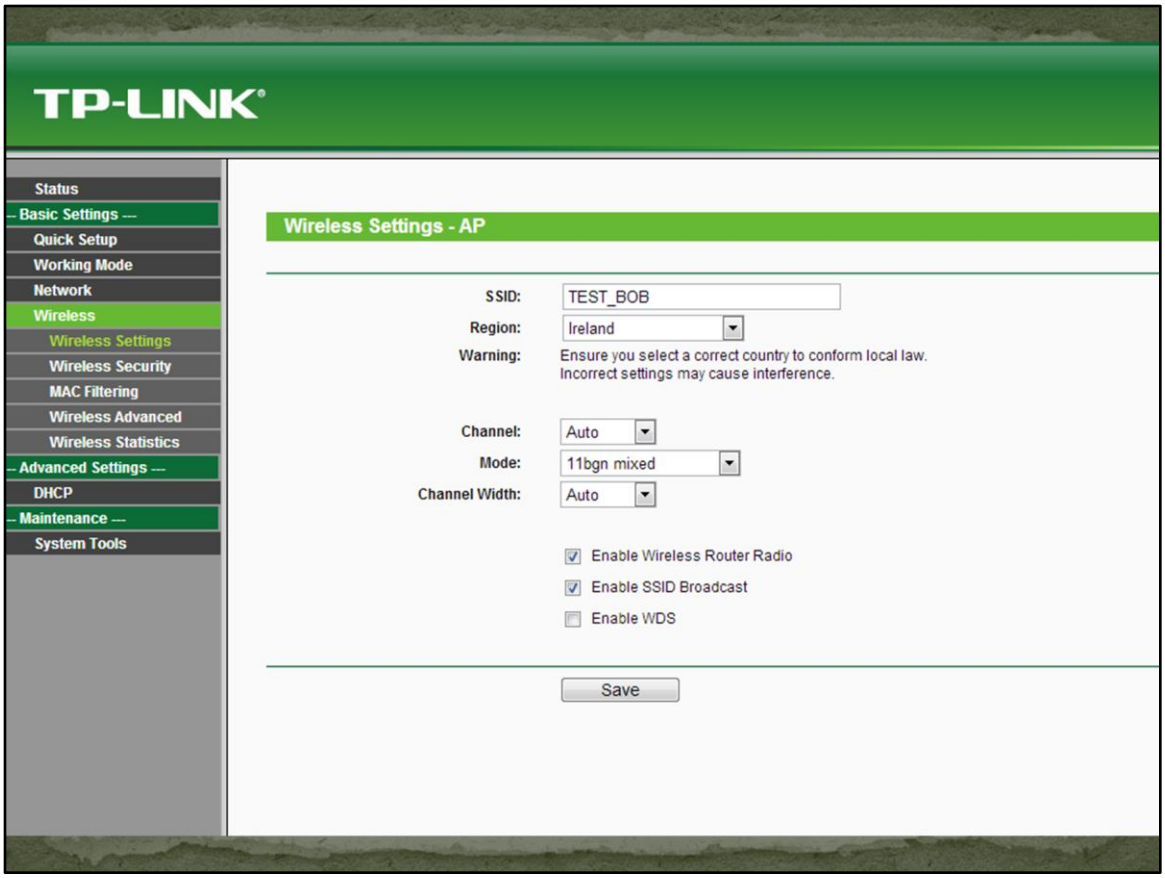
HIDDEN SLIDE

Before we jump in and start running scripts that automate Wifi hacking, we first want to get our lab environment setup, and also get familiar with some of the normal Wifi related tools on linux. To do that we will manually connect to the wifi hotspot. By the way this is probably the only time you'll need to do this ☺

First I get the TP-Link Wifi AP and Plug it in (there is no need to connect up the ethernet port yet). By default it will be running as follows:

•Access Point (AP) mode
•WPA2 PSK Encryption (the password is written on the device)
•Admin panel on 192.168.0.254 with login admin/admin
•DHCP not enabled.

I will also insert the TPLink wifi card into the PWNPad



HIDDEN SLIDES

The first thing we want to do is alter the AP so that it has encryption turned off. We also want to change the AP point name to something memorable.

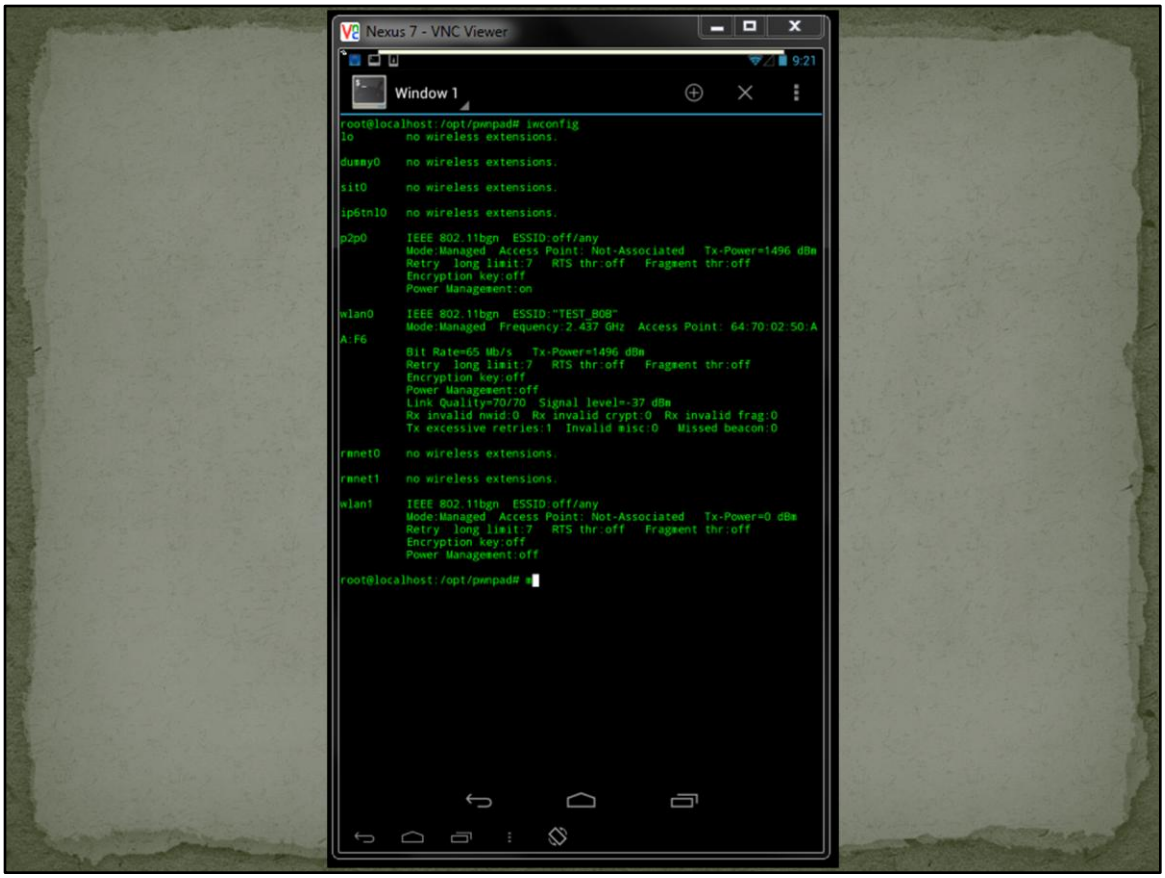
To do this, I connect to the TP-Link AP from my PWNPad (standard inbuilt Wifi) or Laptop, using the password on the device. For your laptop you can connect over wifi, but its easier to just connect via the ethernet jack. As it does not use DHCP you will need to set your IP to something in the 192.168.0.X range, and netmask to 255.255.255.0.

Now connect to the admin panel on 192.168.0.254 with the admin/admin login.

Under Wireless change the name of the access point to TEST_YOURNAME, and you can change the region to the country you are in. Note the country you choose actually has an effect, as we'll talk about shortly. Different countries have laws on what Wifi frequencies and powers are allowed

You can also change the channel. By default most AP are set to channel 6, and if too many others are on that channel it can affect performance. Try and split the classroom between several channels e.g. 1, 6 and 11. By the way, all these "channels" are is different frequency settings

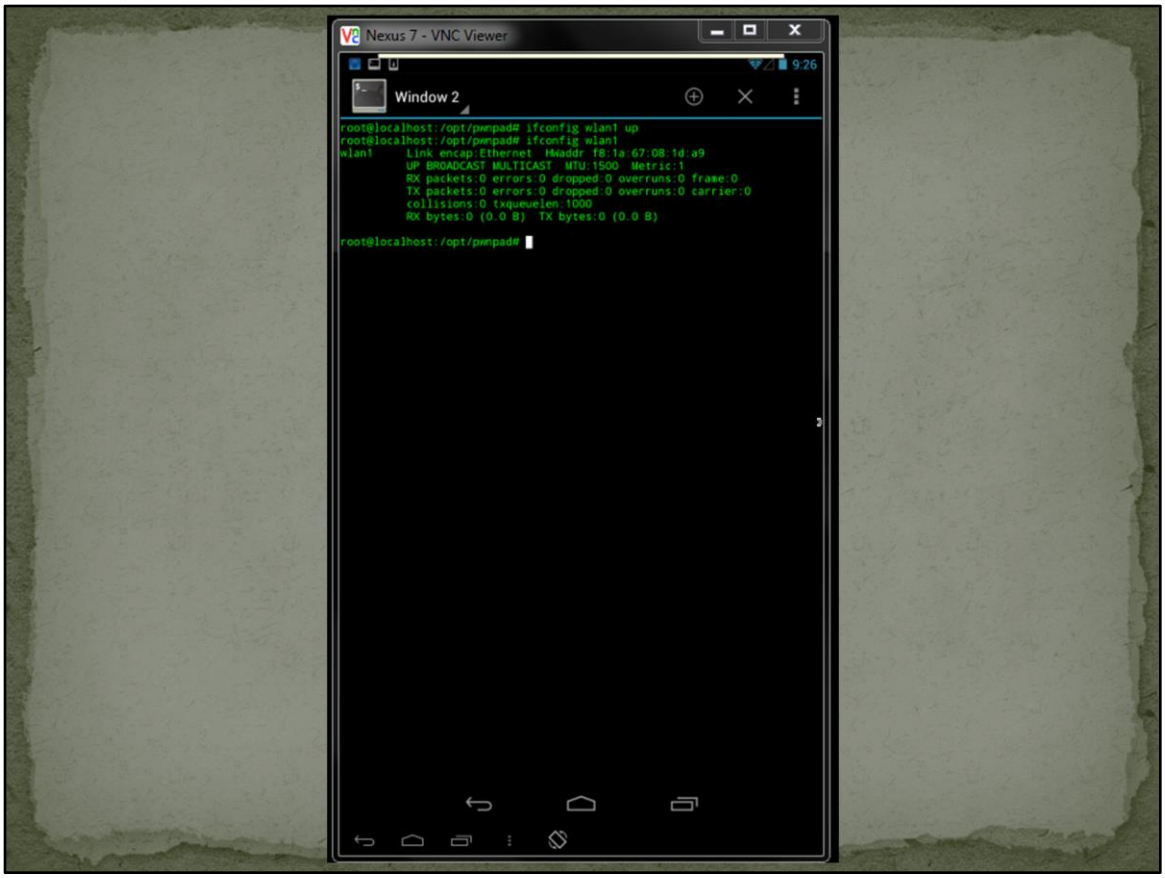
Next go to the Wireless Security page and click "Disable Security" and then choose the option to reboot the device.



HIDDEN SLIDES

Next we are going to see what Wifi adapters are on the PWNPad. Apologies to those of you familiar with Linux, but I wanted to be sure everyone is on the same level.

By running **iwconfig** you will see a list of the Wireless adaptors on the PWNPad. Wlan0 is the internal wifi card, and Wlan1 is the attached TPLink one. You can see that both support 802.11 B/G/N , which network they are connected to (ESSID) and other stats about the card.

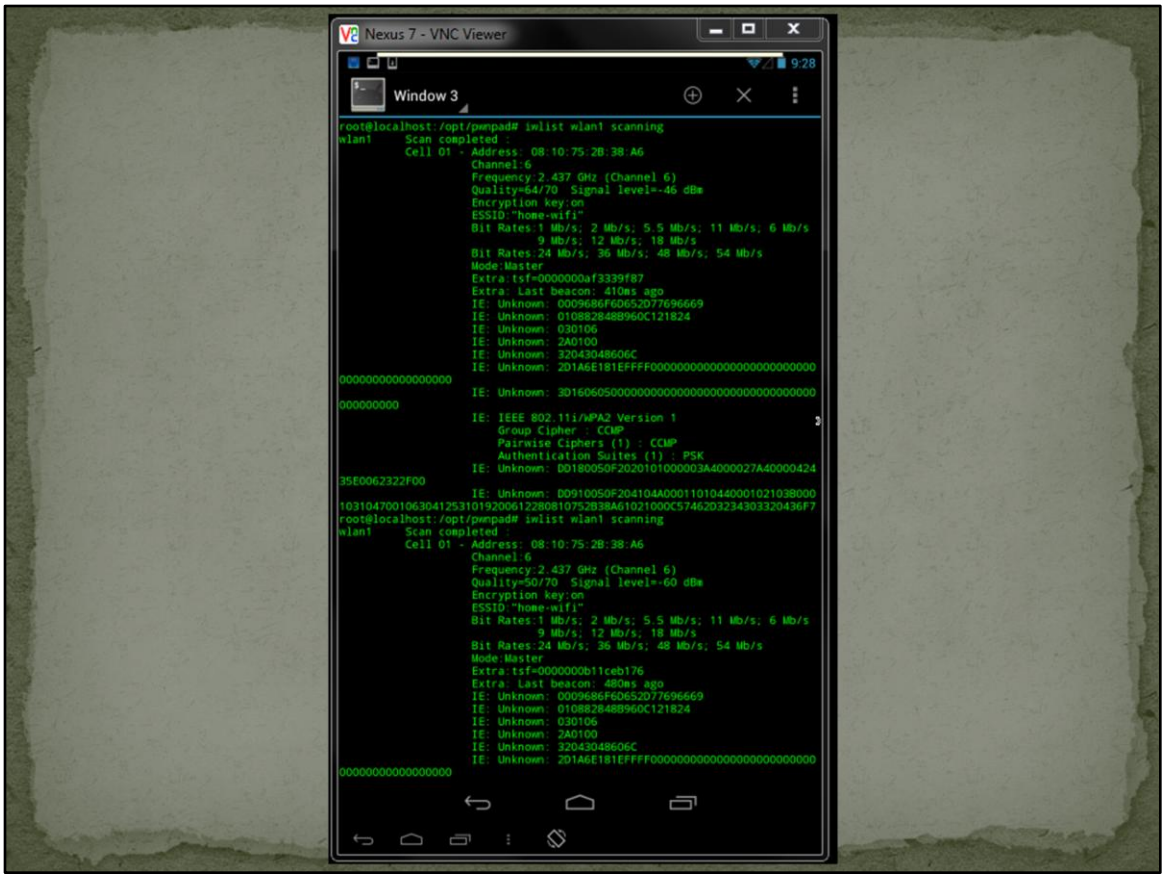


HIDDEN SLIDES

By default when you insert the TP-Link card the PWNPAD should already bring it up as wlan1, but just in case it doesn't you can bring it up running

ifconfig wlan1 up
ifconfig wlan1

(Note: You can tab complete to speed up typing)



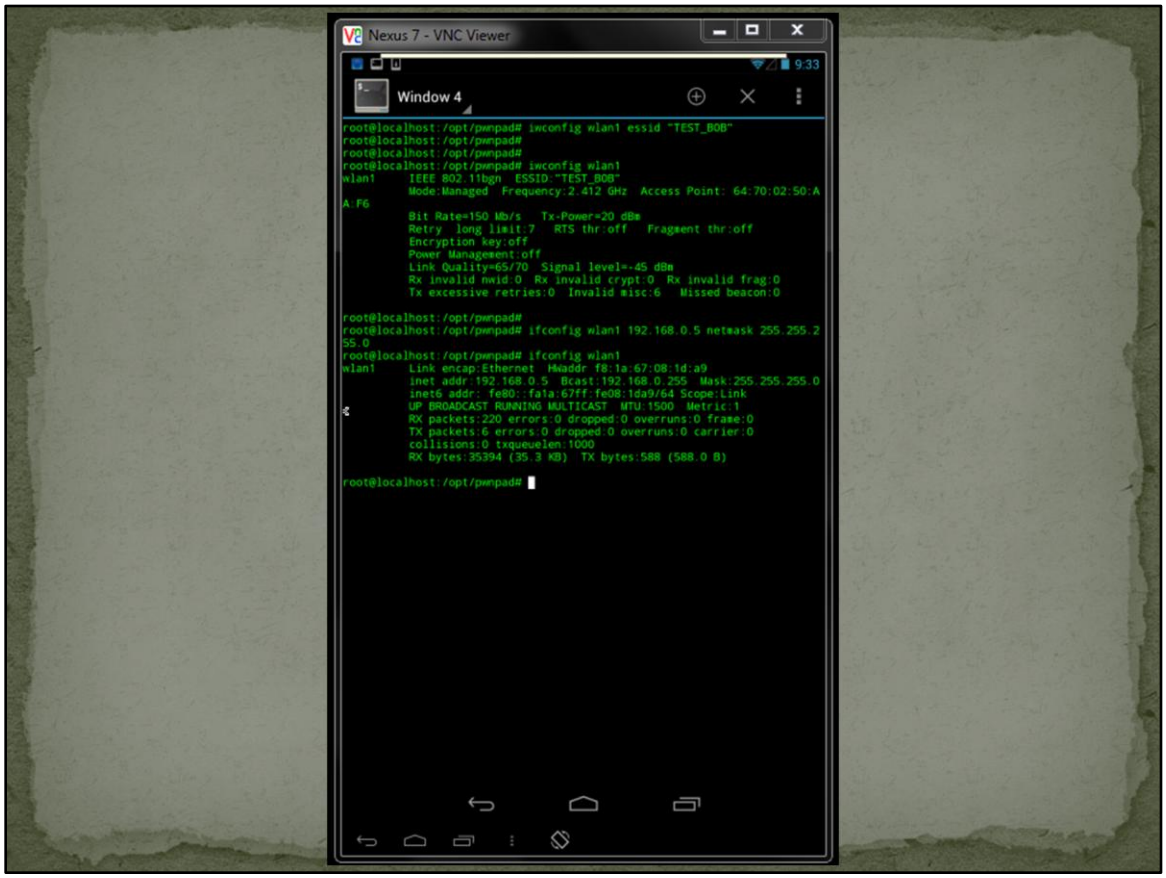
HIDDEN SLIDES

The PWNPAD has a bunch of graphical tools to scan for available wireless networks, but its useful to have done it manually at least once. Simply run

iwlist wlan1 scanning

Each AP is detected as a Cell and it gives loads of details for them, such as:

•MAC Address
•Channel & Frequency (Note each channel has a specific frequency)
•Quality / Power
•Encryption Type
•ESSID



HIDDEN SLIDES

Lastly to authenticate to the Wifi AP run:

iwconfig wlan1 essid “TEST_BOB”

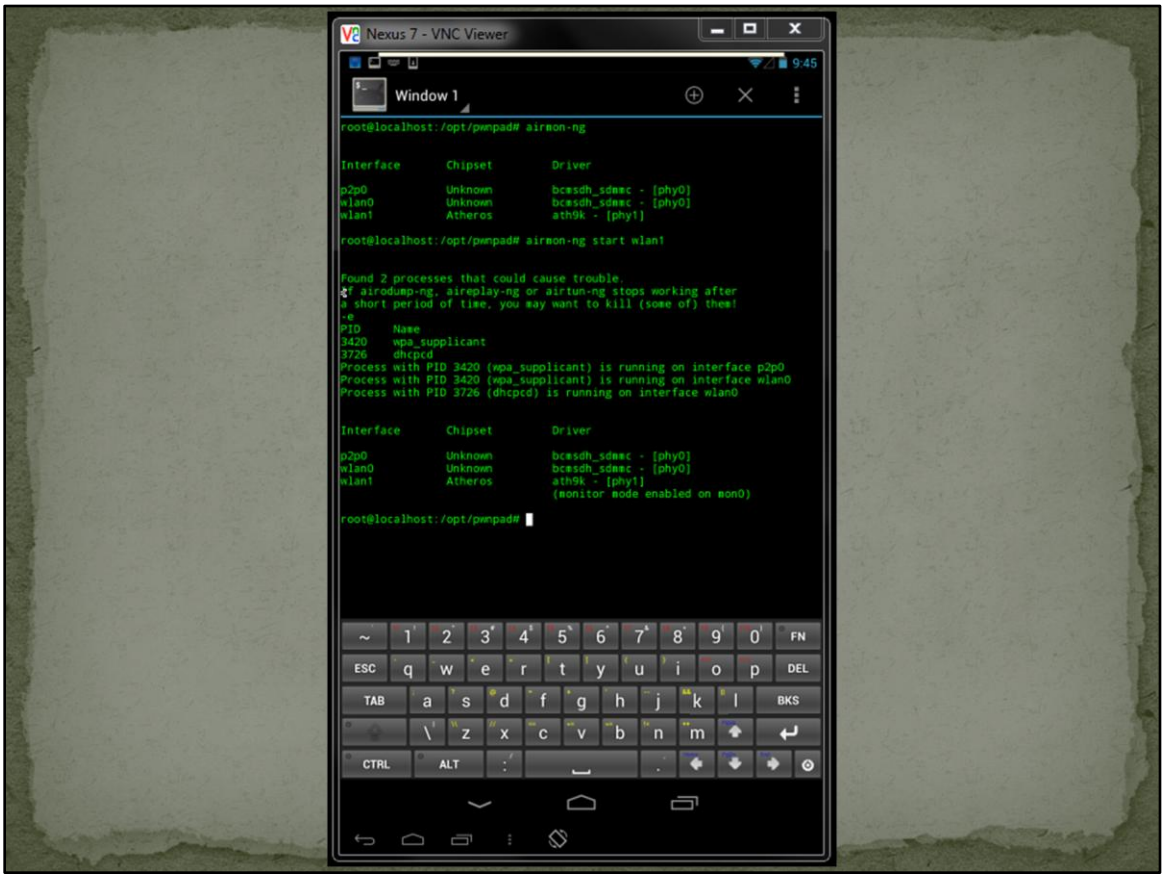
(change to your ESSID name obviously). You can now see wlan1 is connected by running iwconfig

Then as DHCP is not enabled you can set an IP and netmask manually

ifconfig wlan1 192.168.0.5 netmask 255.255.255.0

How ifconfig should show the assigned IP – bingo, you are on the network!

Ok – so next we are going to do some basic sniffing – before we move onto the fun part of breaking encryption ☺



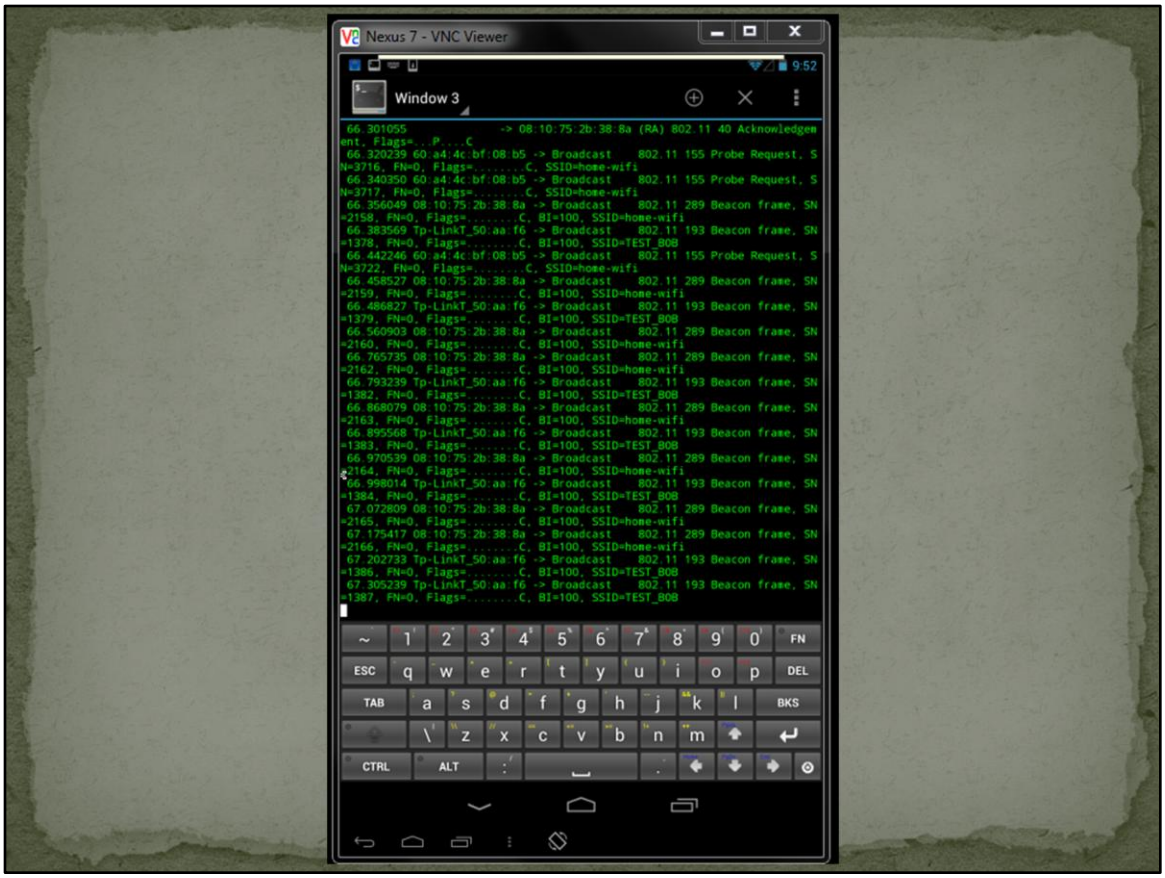
LIVE DEMO

On a wired network when you want to sniff the network you put your card in promiscuous mode, on a wireless network you put it in Monitor mode. Monitor mode lets the card see all traffic received from the wireless network, even packets not destined for the PWNPad. Unlike promiscuous mode, monitor mode allows packets to be captured without having to associate with an AP first. Monitor mode only applies to wireless networks, while promiscuous mode can be used on both wired and wireless networks.

Simply run

Airmon-ng start wlan1

This will create a new interface called **mon0** which you can see details on using ifconfig



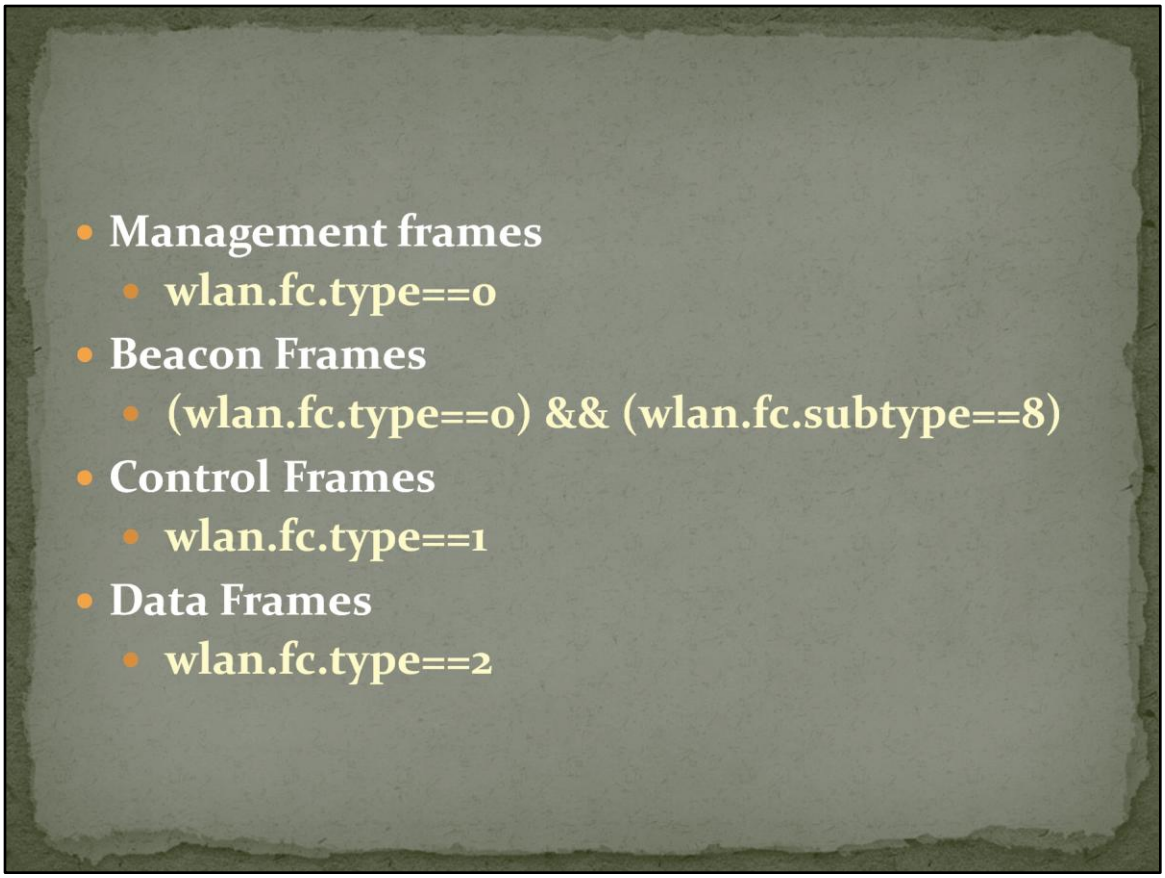
LIVE DEMO

To capture traffic into a PCAP format the handiest tool on the PWNPad is to use the **tshark** script, choose mon0 and the source and choose to write out the log.

If you only want to capture traffic from a specific channel (e.g. 1) you can do that easily by locking the mon0 interface into a channel using:

iwconfig mon0 channel 1

There are no really good PCAP viewers for Android that I've been happy with so far, certainly not at the level of WireShark – but you can copy the PCAP file off later for analysis. In wifi pcaps there are three main type of packets.



Another important note on wireless sniffing. WLANs typically operate in three different frequency ranges – 2.4GHz, 3.6GHz and 5.0GHz. Not all cards support all ranges – so you need to have the right one for the job.

Another thing to know about Wifi sniffing, in each of these bands there are multiple channels – but your wifi card can only be on one channel at a time. Think of this like a Car Radio. You can tune it to only one channel at a time, if you want to hear something else – you change the channel.

This same issue applies to packet injection which we'll do later on – you can only inject into the channel you are locked onto

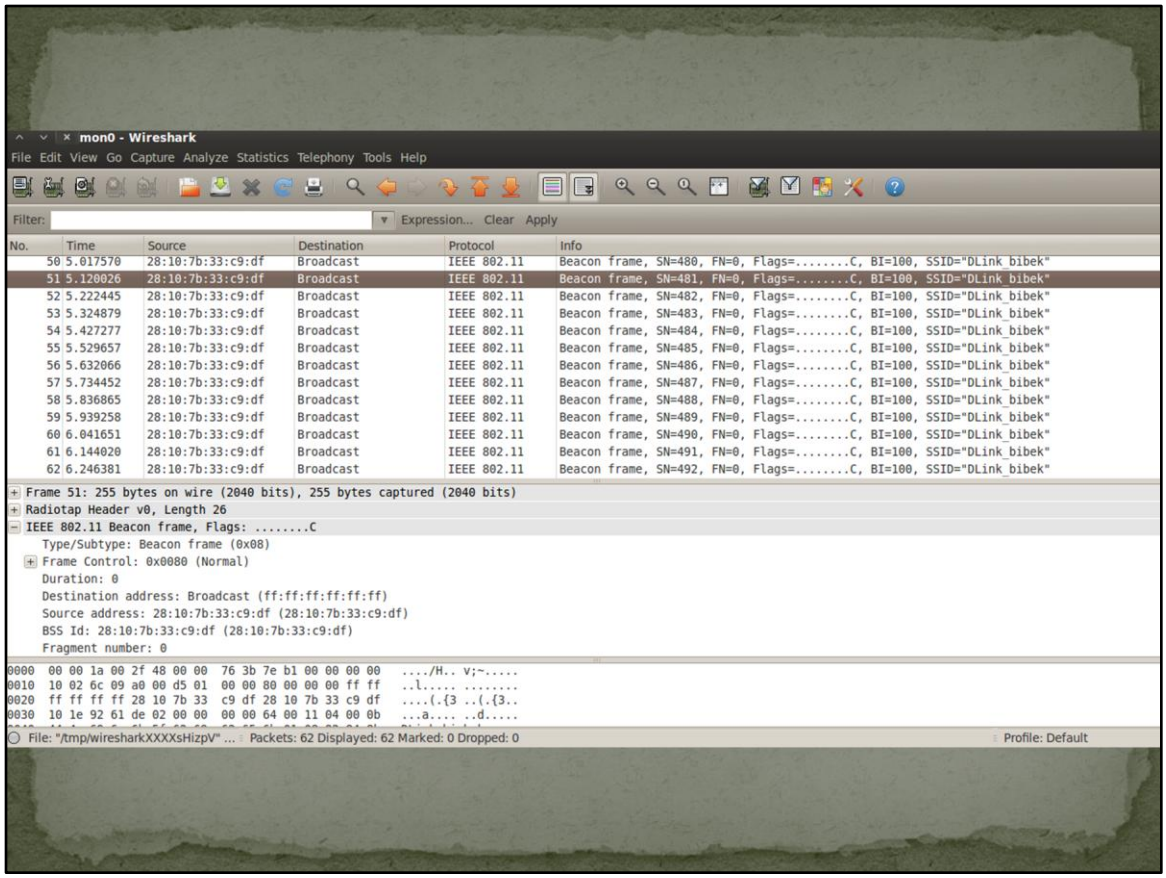
Also of course you can only sniff packets that are in range of the antenna you are using. That means that for some Access Points on the edge of your range you may not get all packets. Likewise you may see all the communication from an AP to some client attached to it, but no responses. This could happen if the AP is in range, but the client is not

So sniffing is pretty straightforward – but if everything is encrypted it does not help you much

We have a few more small things to cover before we go onto actual wireless cracking.

There are 2 very common methods of “securing” wireless connections carried out by poor IT admins - that you will still come across today - which are all trivial to defeat...

- Hidden SSIDs
- MAC Filtering

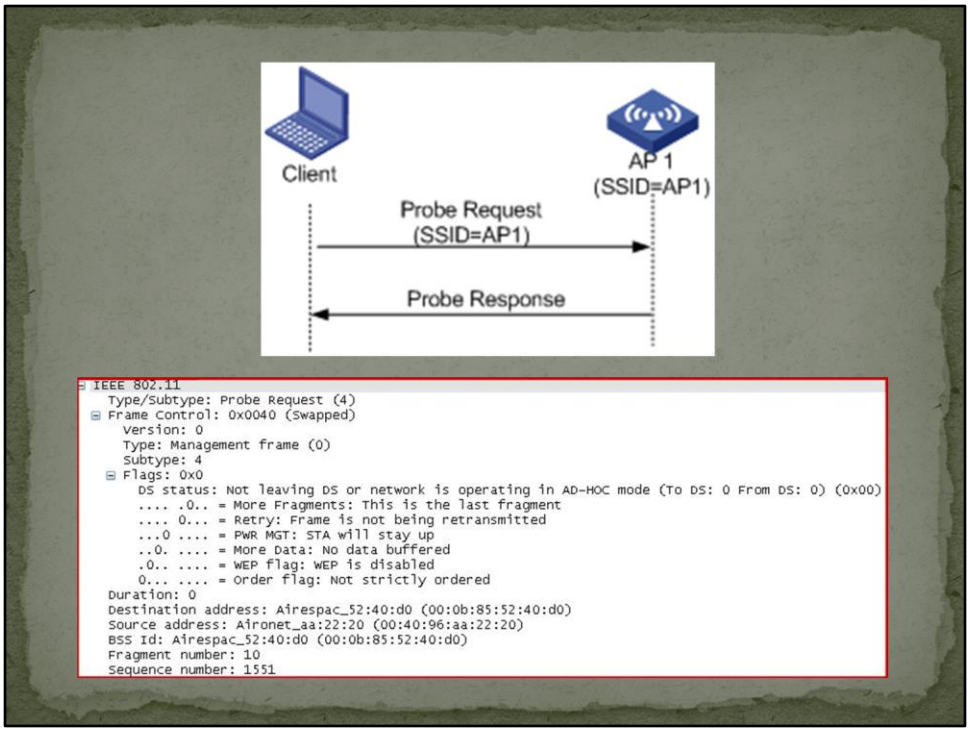


Its amazing how many guides to securing Wifi APs still advise you to hide your SSID as a form of added security. The only thing this tells you is that the person writing the guide has absolutely no idea how the 802.11 protocol works.

Have a look at the slide, this is what a normal Access Point looks like, sending beacon frames several times a second announcing its SSID – and this is what is shown on your phone, tablet etc.

Source	Destination	Protocol	Info
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2863, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2864, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2865, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2866, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2867, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2868, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2869, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2870, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2871, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2872, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2873, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2874, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00
28:10:7b:33:c9:df	Broadcast	IEEE 802.11	Beacon frame, SN=2875, FN=0, Flags=.....C, BI=100, SSID="\000\000\000\000\000\000\00

And here is an Access Point that has been configured to not broadcast the SSID. To connect to this Access Point you will need to know the real SSID, otherwise you will get an authentication failure. That may seem like added security – but lets take a look at how an association handshake happens in 802.11.

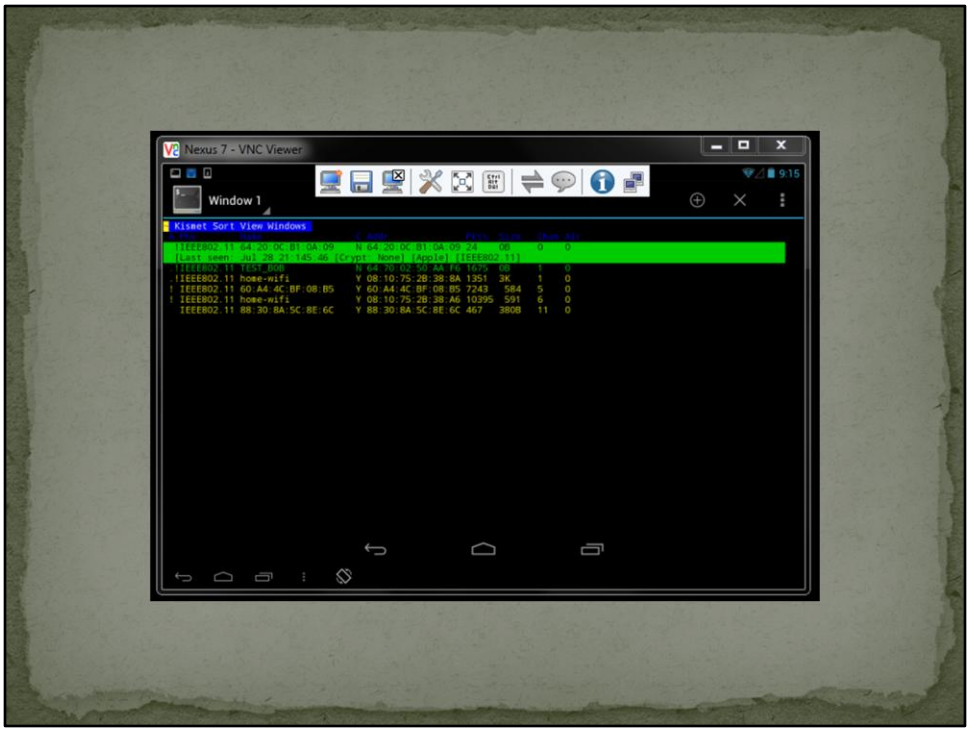


When a client connects to an Access Point it send a packet called a “Probe Request”, and the AP responds with a “Probe Response”

Here is a sample of a probe Request packet – can you see the issue?

The BSSID (AKA Access Point name) is sent in the clear over the air ☹ Anyone in range to sniff it can find the Hidden SSID as long as some client is connected to the network.

If you come across this sort of setup yourself there are lots of ways around it. One way would be to simply use Tshark or Wireshark to capture the packets until some new legitimate device joins the AP, but there are better tools. In particular most people use either **Kismet** or **Airodump**



For now I'll use Kismet, to let you get familiar with it (and because we'll use Airodump in the next bit)

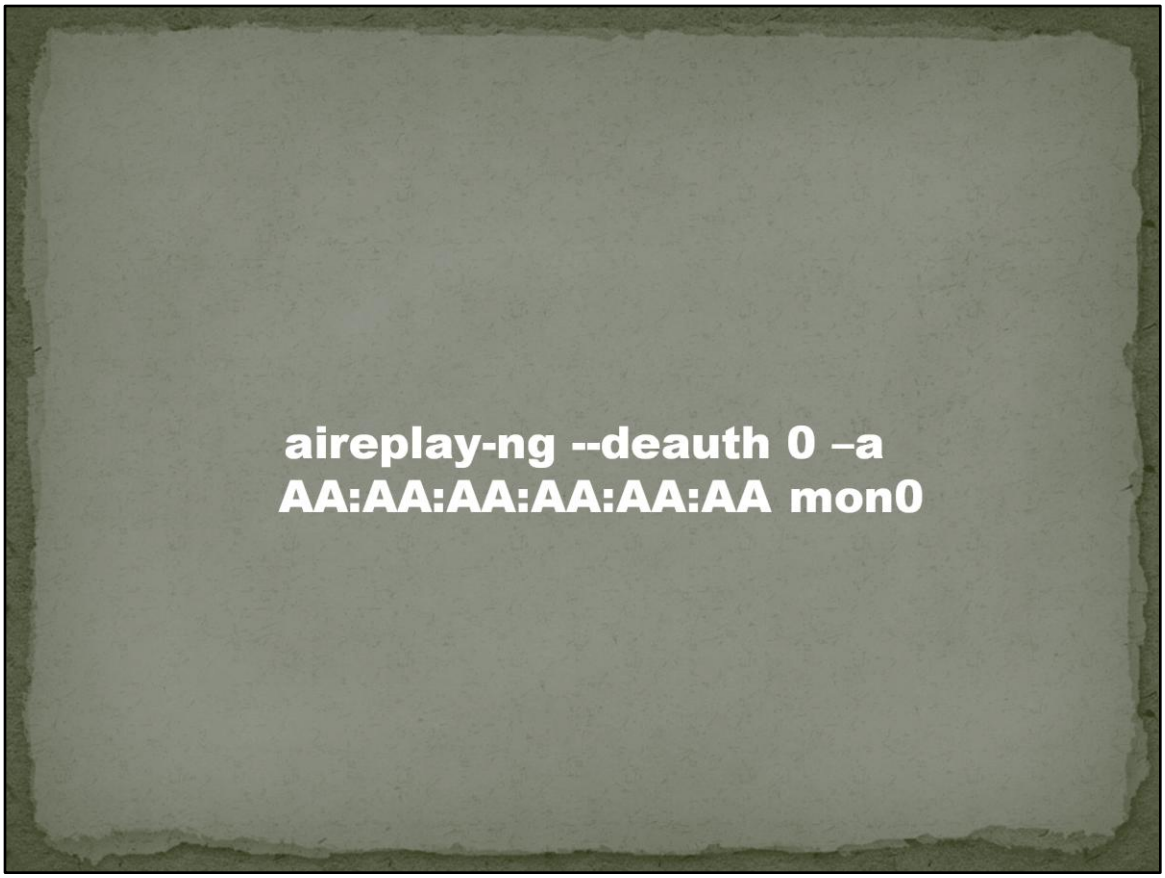
I start up Kismet. You'll notice the layout is a bit messed up on the tablet (but will be fine on Kali), this is just a draw back of using Kismet on a tablet – its does not look like this under Backtrack for example. In a moment we will be changing the view a bit to help tidy it up.

You'll be asked:
•Is it ok to run Kismet as Root
•Automatically start Kismet Server
•Start Kismet Server

Press OK for all of those. You'll then end up on a rolling log screen. Press **TAB** and **Close Console Window** to enter the main Kismet screen. It will look like a bit of a mess.

To access the Kismet menu press either **ESC** or **~** . Now using the cursor keys will let you move around the menus (which still look a bit messed up). Go to the view menu and ensure only **Display Wireless** and **Device List** are selected. Once you do that it should look more readable, like on the slide. The columns for Kismet should be fairly self explanatory and pressing enter while on one of them will give you more details. To get out of this window once more press Esc or ~to access the menu.

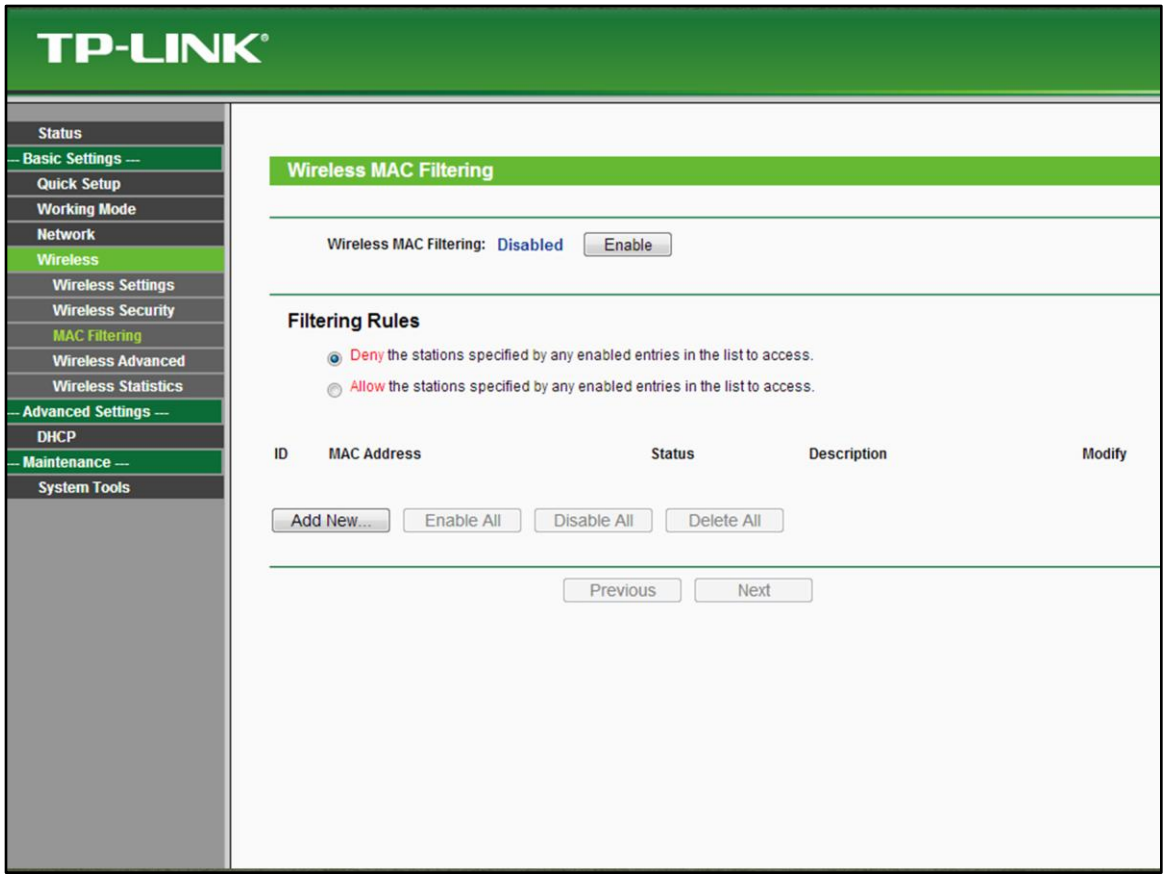
If there is a AP with a Hidden SSID as soon as a client connects to it Kismet will update the description with the SSID seen in the probe request.



What happens though if there is a client connected already to the AP you are interested in. You would need for it to disconnect and reconnect before you see a probe request ☹ Of course wifi hackers are not noted for their patience.

Have a look at the command on the slide. Aireplay is a tool we will be using a LOT. It allows you to inject 802.11 frames into a network, even one you are not authenticated with! For this command you are sending a DeAuthentication packet to all stations (aka clients) on the AP with BSSID AA:AA:AA:AA:AA:AA using the network interface mon0. So simply replace the MAC address here for the one used by the AP

Note you can only do this with the Tplink card, as the Atheros chipset supports the packet injection, but the internal card does not



MAC Filters

Most Wifi APs will let you implement MAC Filtering. Either you can set up a list of denied MAC addresses, or you can setup a list of allowed devices.

Getting around the Denied MAC addresses is REALLY easy on PWNPAD or Linux. Just run the HostMACChanger script, and hey presto. Incidentally this on its own will get you infinite free wifi at lots of airports and hotels that give you a fixed time amount of Wifi access for free before charging you (along with clearing your browser cookies for their login pages, or using something like Chrome Incognito mode).

To get around Allowed mode is also really easy in the wireless world. All of those packets are travelling by in the clear, and you can sniff them to find a MAC of a device that is authenticated to the network already. Even if encryption is being used it won't matter – the encryption is of the data only, the MACs are at layer 2.

If you where running a sniffer while trying to connect with a banned MAC you will see several Authentication Failure messages.

The tool we'll use to see clients associated to Wireless APs is **Airodump**, which you'll find on the main screen.

By default it will listen on mon0 and will channel hop across all the channels until it identifies.

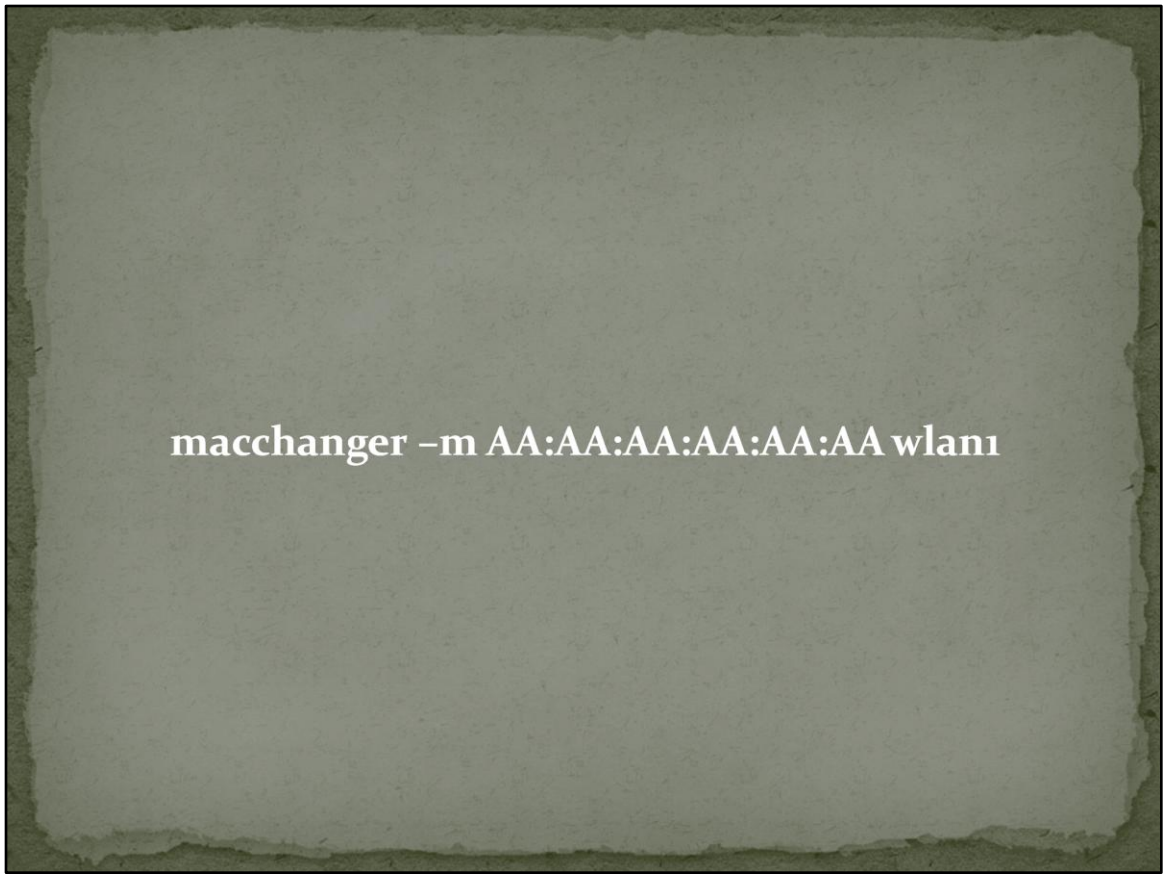
The airodump layout is in two parts – the top shows the available Access points, and the bottom any wireless clients. For each AP you will find the BSSID (the APs MAC), the PWR signal, number of beacons it has sent (that we have seen), number of data packets sent, Channel, Encryption type and ESSID (AP name)

For clients you will see the AP it is connect to (if any) as a BSSID, the clients MAC (called Station here), and then Power etc, as well as the network name shown in the probe request. You will also see some clients for which the BSSID reads **"(not associated)"**. This is because most wireless devices such as phones, tablets and keyboards are constantly sending out probe requests to all of the saved wifi configerations they know – even when the AP is not in range.

If the output is showing too much information, because you are in an area with a lot of wifi activity, you can manually run airodump from the rootshell but limited to just a certain AP:

airodump-ng –c 1 –a –bssid 01:02:03:04:05:06 mon0

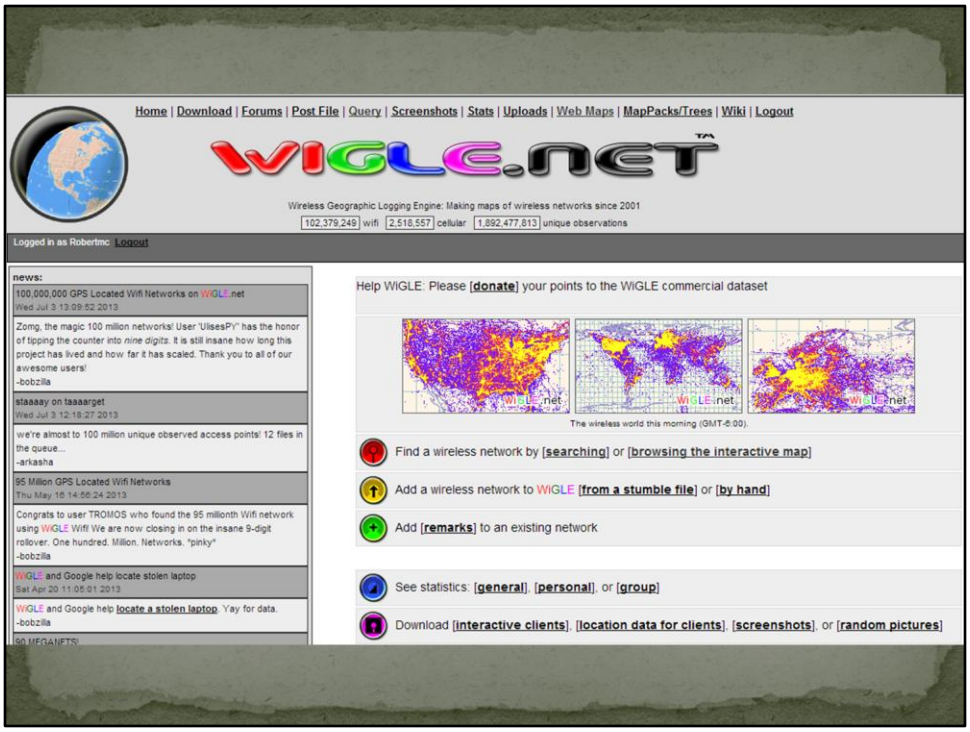
-c	**Channel**
-bssid	**AP to link to**
-a	**Client section shows only associated clients (not probes)**



Once you have discovered a MAC address that is allowed (because it has been seen to be associated with the AP by Airodump), you can simply change the MAC of your wifi card by running

macchanger –m AA:AA:AA:AA:AA:AA wlan1

Where AA:AA:AA:AA:AA:AA is the allowed MAC. Note the wifi card can't be up when you are doing this, so you may need to do a simple **ifconfig wlan1 down** first. Macchanger is the same tool that the HostMACChanger script calls.



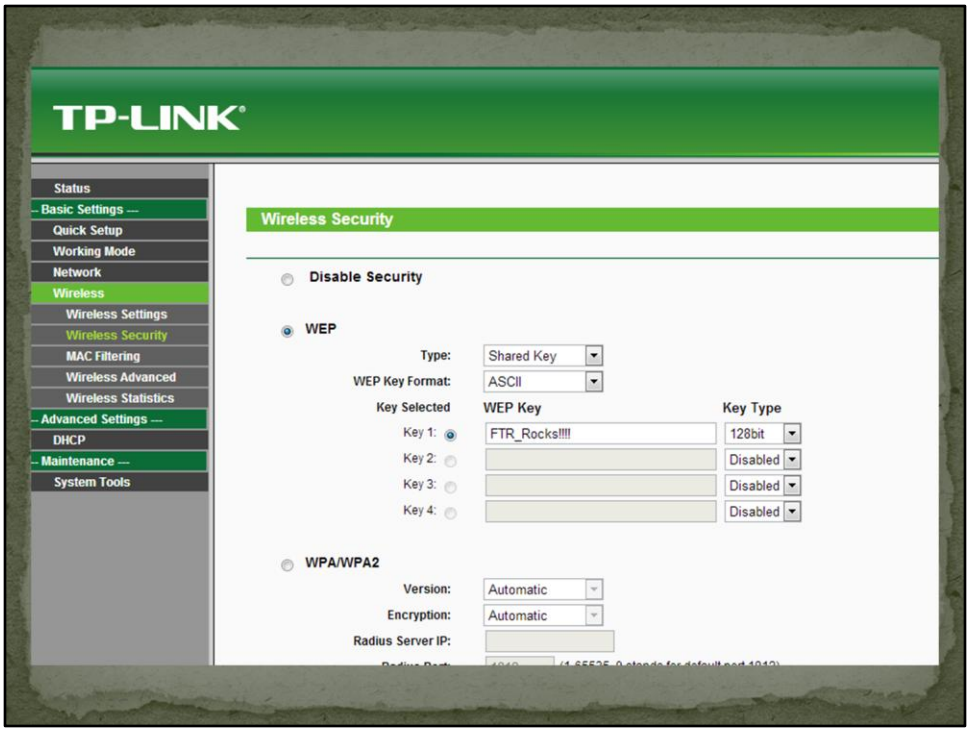
Incidentally probe requests have another major issue, especially when they are for fairly uniquely named networks. If you not down what networks someone is probing for you can use the site **Wigle.net** to search for those APs. This can be used to geolocate where someone works or lives, and even things like conferences they have attended (e.g. Probing for BHVegas2013)

Wigle is a site where wardrivers can upload their logs so that others can search over them. There is also an Android app for Wigle which you can put on the PWNPad that will record details on all APs in the area you are travelling to along with GPS coordinates, and lets you easily upload the data.

You need to sign up for a free account to be able to use the search feature, but can also browse the interactive map

Ok – thats the trivial stuff out of the way. Now onto the really fun stuff ☺

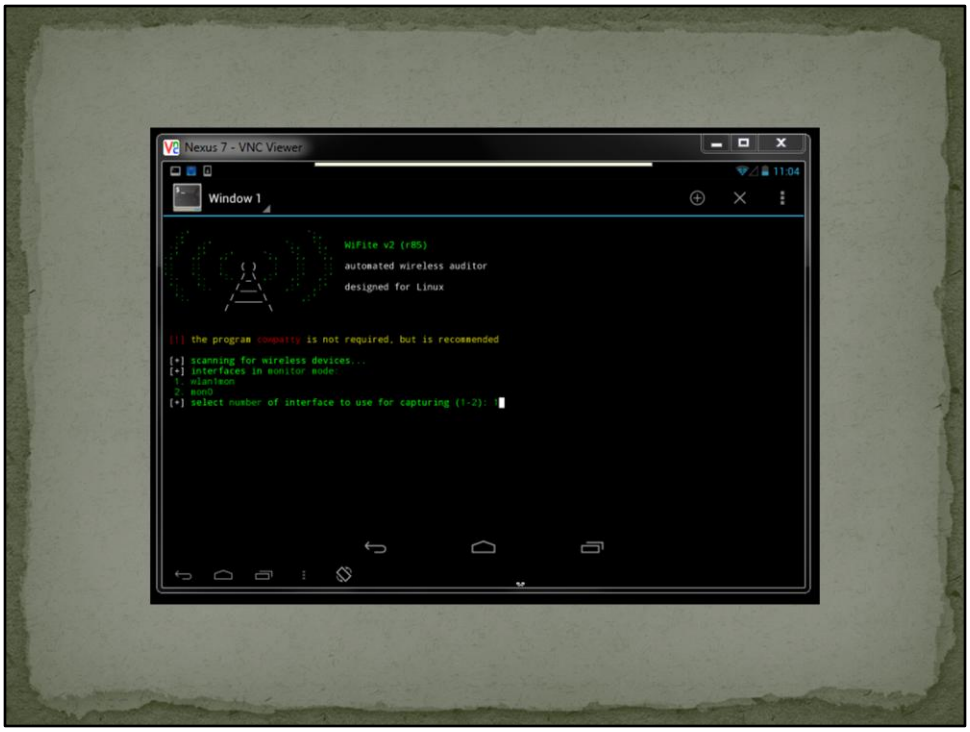
Next we are going to crack WEP and WPA, starting with WEP.



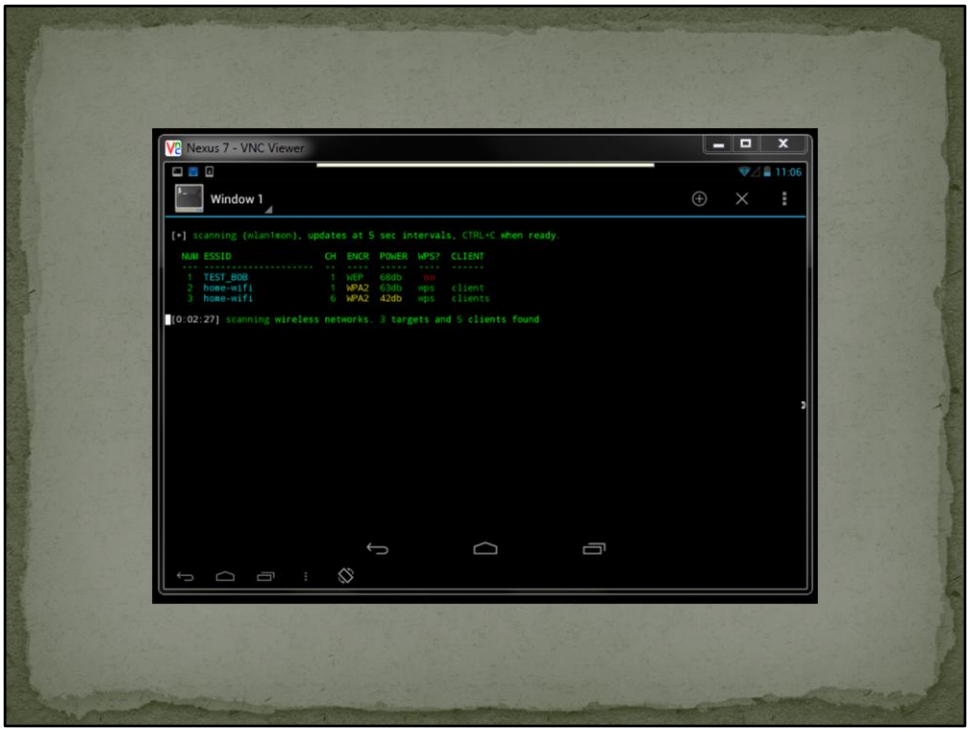
WEP has been known to be flawed since 2000, but surprisingly its still in use today. When Pen-testers or hackers see it they weep for joy. Its worth nothing that there is not a single crypto flaw in WEP, there are lots! I'm not going into the details of the crypto flaws here, instead we'll focus on the tools to break it.

For a first step we'll connect to the AP and set it up to use WEP with a key of your choice. I went with a 128 bit key with ASCII value "FTR_Rocks!!!!"

Now we connect to the AP with a non-PWNPad client. Instead of going through each of the steps manually, lets jump straight to the fun bit and crack it using the automated WiFite script. We'll then go through each of the steps in more detail so you understand whats happening at the low level.

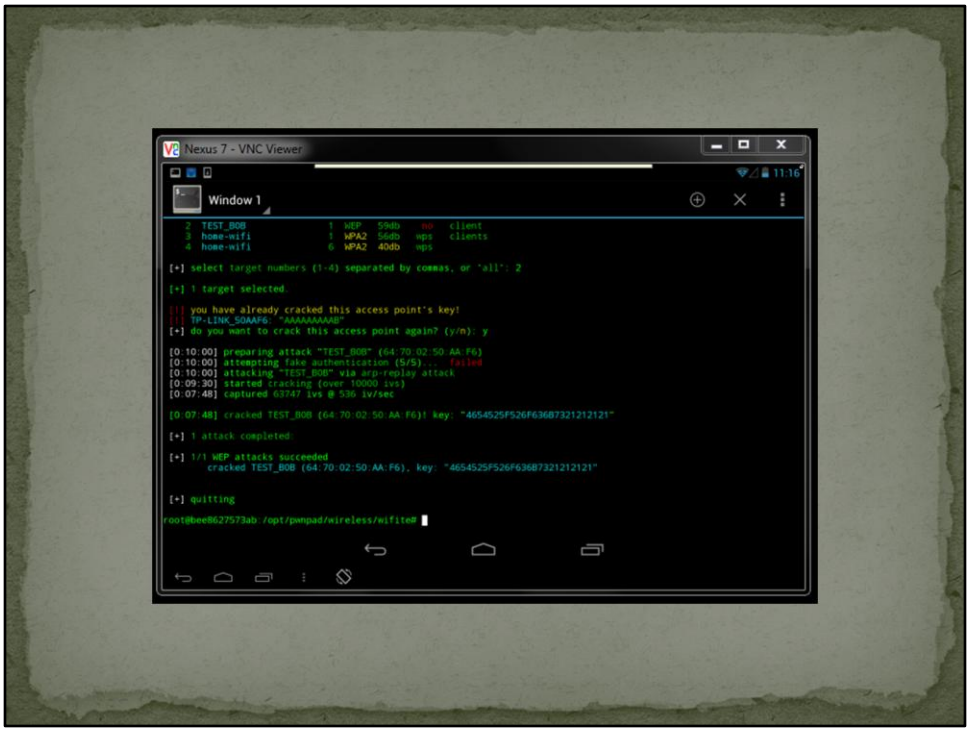


Wifite will setup a new monitor mode interface for Wlan1 called **Wlan1Mon**, you can select this for the attack



Next it will show you the available APs . Press VolKeyDwn+C (CTRL+C on Kali) when you see your AP in the list, and then select it.

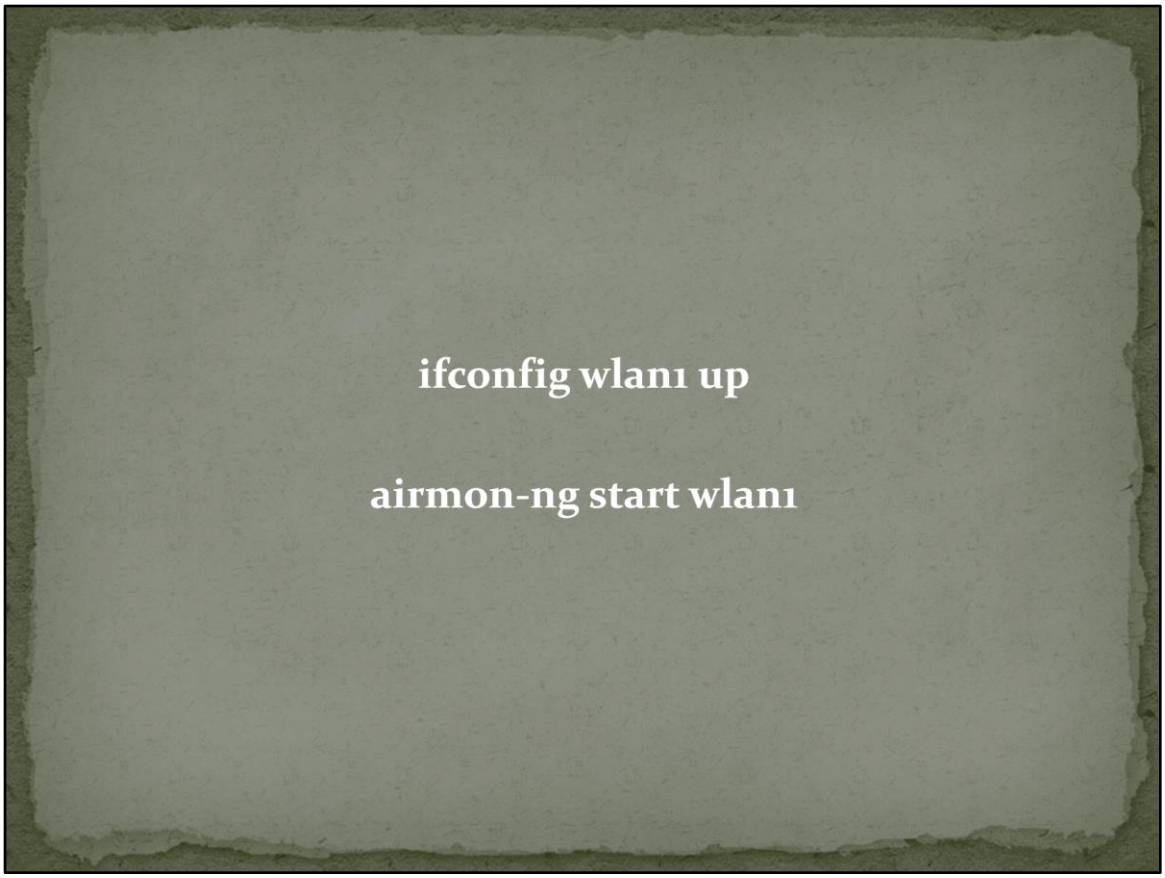
After that Wifite takes care of everything else. What wifite is actually doing at a high level is capturing packets to and from the target AP that contain what is known as Initialization Vectors (IV). Once it hits about 10000 of these it will start to crack them in parallel while still capturing packets. These IV packets are normally only sent during the authentication process, so to speed things up Wifite will deauth a client repeatedly from the network.



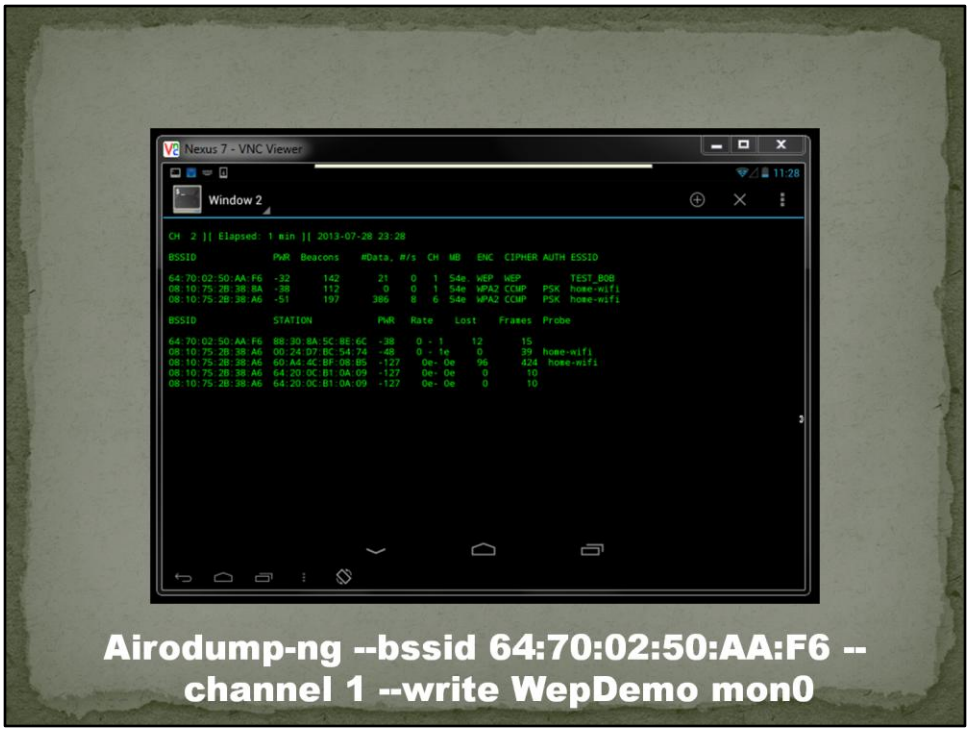
After a very short period of time (in this case 3 mins) – hey presto, you've cracked your first wireless network ! The key is displayed in HEX as opposed to as ASCII, but you can change it back with any HEX to ASCII convertor.

Now that we have done it the automated way, lets go through what the Wifite script is automating by doing it manually. I'll show this once in slides, and then put everything on one slide and try it out

We start by leaving your AP setup as before and our other client connected.



Just for cleanness disconnect the wifi adapter and re-connect it, then we will run both commands on the slide. This will create a mon0 interface in monitoring mode, which you can confirm with iwconfig.



Use the airodump shortcut and note down the SSID, Channel and MAC of your AP.

Then exit out and run the second airodump command on the slide

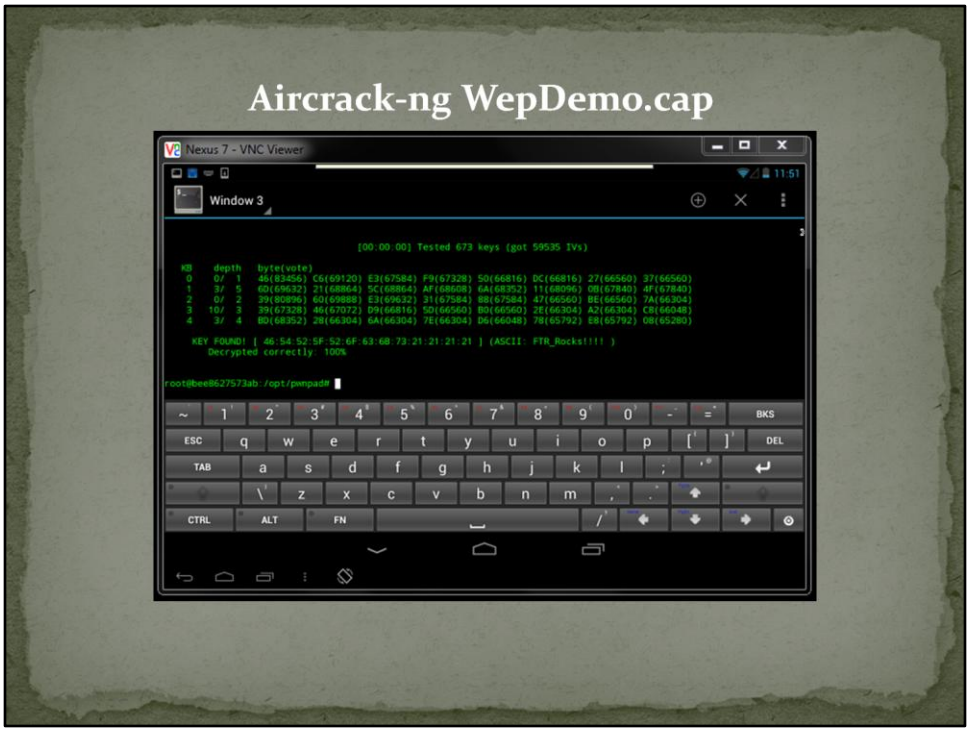
You should now see something like on the screen. Next we want to deauth that client to capture at least one AUTH exchange. We can do this in another rootshell window. Once you've got at least one you can stop.

You can have a looks at the captured pcap files so far by running **ls** in the main folder. You'll notice the number of data packets is still low, and we need quite a lot for a good WEP crack – so we will use aireplay to forge some packets. In particular we use aireplay to capture ARP packets and replay them a few thousand times.

Even though aireplay does not know the key it can still identify ARP packets as they have a fixed header length.

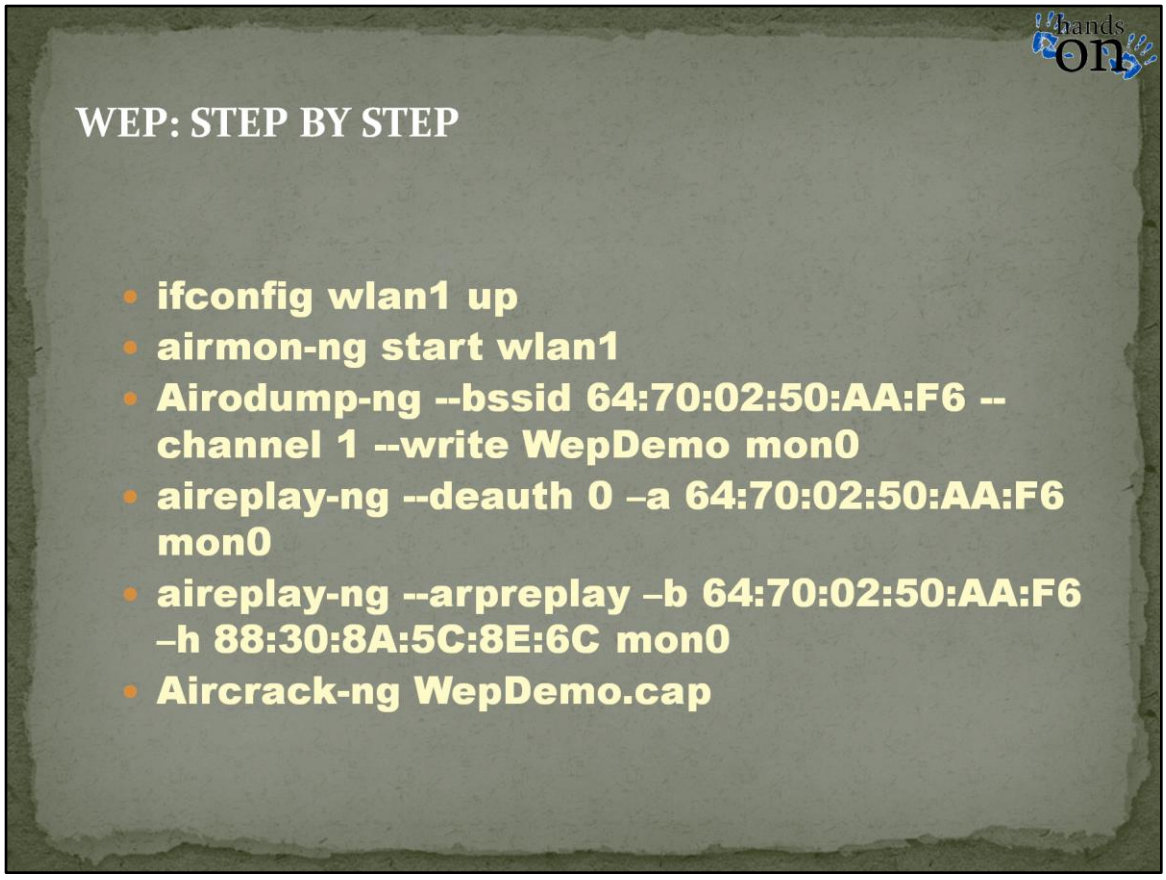
Running the aireplay command on the slide will tell the PWNPad to capture a ARP request from the client (-h) to the AP (-b) and replay it indefinitely.

If you swap back to the Airodump window (swipe left / right) you should see the number of data packets rocket up. Once it hits about 10K its time to start cracking.



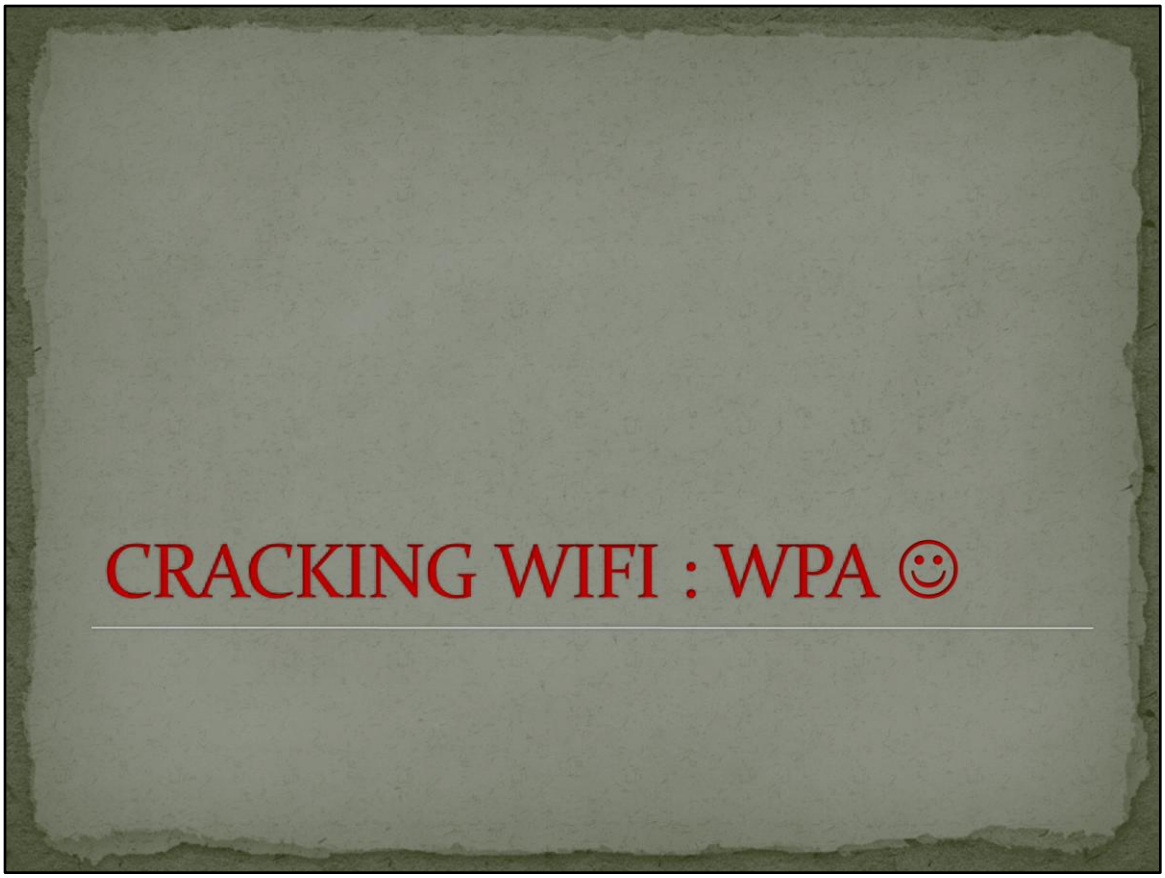
Simply run Aircrack on the pcap file that you are capturing to. The exact number of packets that will be needed to successfully crack is non-deterministic, but Aircrack will go through all of the packets in the pcap before reloading it automatically and trying the new ones. It should take no longer than 5-10 mins at the most however.

Once its done – hey presto, it appears on the screen ☺



DEMO TIME

Replace with your MAC addresses obviously ☺



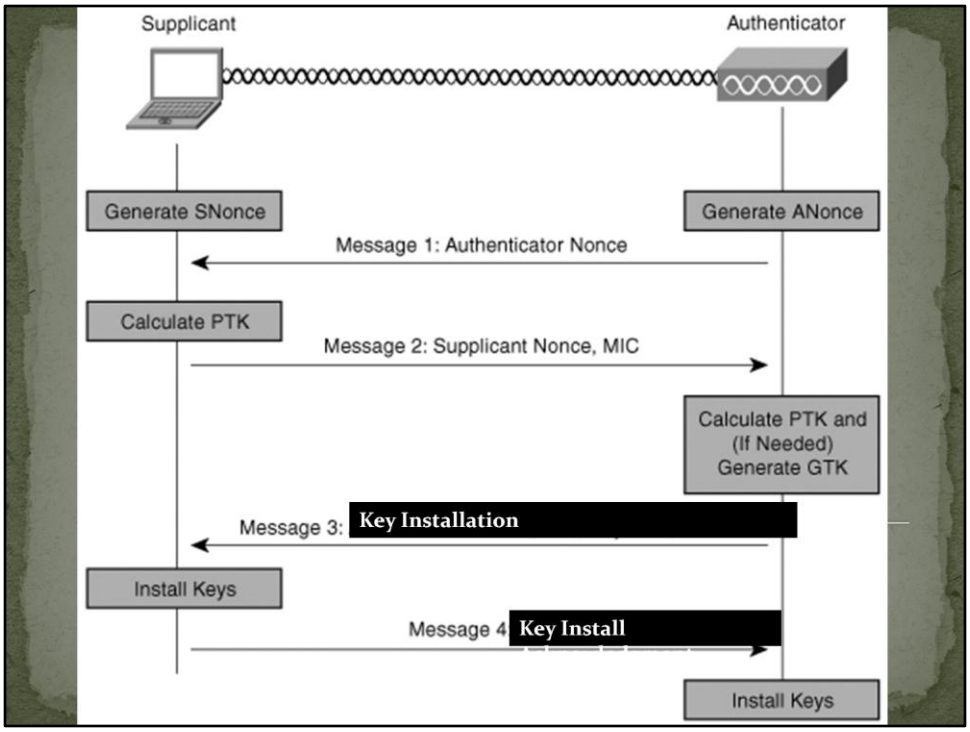
Now that you have seen how easy it is for an attacker to crack WEP, lets move onto WPA

WPA stands for Wi-Fi Protected Access. There are two versions – WPA1 (or just WPA), and WPA2

WPA1 uses the TKIP algorithm for encryption, where WPA2 uses the much stronger AES-CCMP algorithm. The idea behind using TKIP in WPA was that it could be implemented via a firmware upgrade of existing WEP cards. WPA2 needs different hardware however.

Both allow for either a Pre-Shared Key for authentication (called Personal mode) or EAP-based authentication with a radius server (Enterprise). For now we will look at the Personal version.

The main way to attack either form of WPA is via a dictionary attack. To understand it better lets start by looking at the WPA four-way handshake, as all parts of this must be captured for an attack.



The Wireless client (aka Supplicant, Station, etc) first carried out a normal **Association** with the Access Point using Probe Request / Response. Next comes the **Authentication**

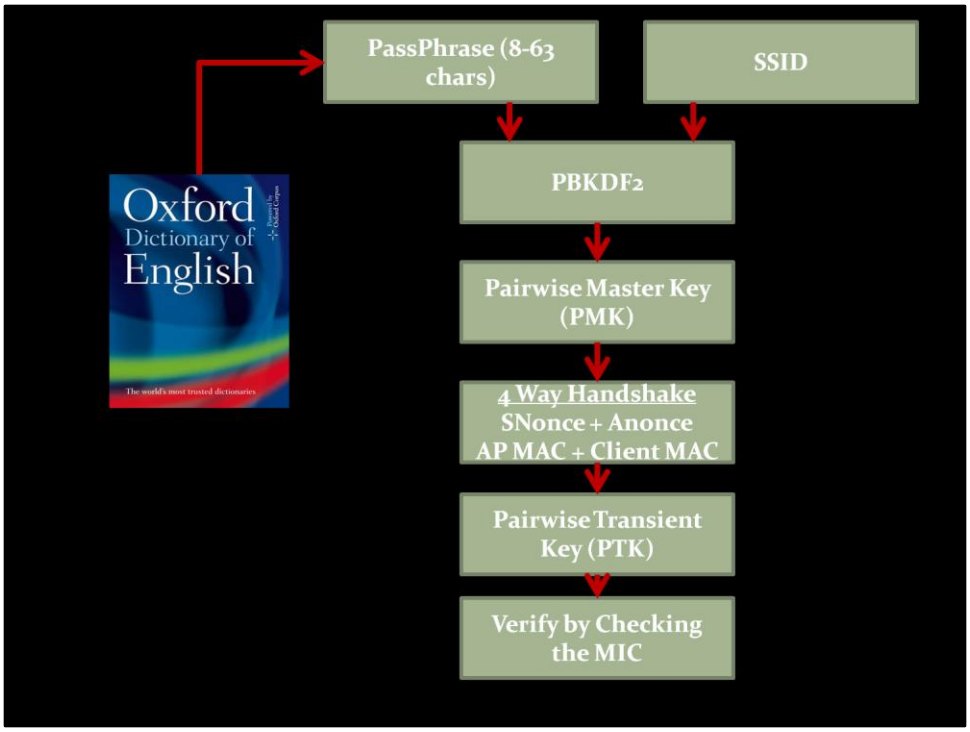
Both the client and the AP have a shared encryption key, which was shared through some out of channel communication. This key is called the Pre-Shared Key, or more correctly the Pairwise Master Key (PMK)

The AP generates a nonce which is unique for this PMK and is there to prevent replay attacks. The client generates its own nonce and uses both nonces together with the PMK to generate a Pairwise Transient Key (PTK) – essentially a key unique for this connection. It then sends this 2nd nonce to the AP so that it can do the same. The MIC is a special hash that allows the AP to know the PMK was used.This is also why have a easy to remember WPA key in a cafe is much better than having an open AP – everyone in the cafe is using the same PMK, but different PTK – so can't read each others communications.

In total to calculate the PTK you need 6 parameters - both nonces, the MACs of both the client and AP, the SSID and the PMK. An attacker eavesdropping on the authentication can get 5 of those parameters, essentially everything except for the Pairwise Master Key. So how is that key generated?

Well the PMK is derived from the shared passphrase to access the network, and the network SSID. Both are combined via the Password Based Key Derivation Function (PBKDF2) which spits out the 256-bit shared key

Src:
http://etutorials.org/Networking/Wireless+lan+security/Chapter+8.+WLAN+Encryption+and+Data+Integrity+Protocols/Key+Management/

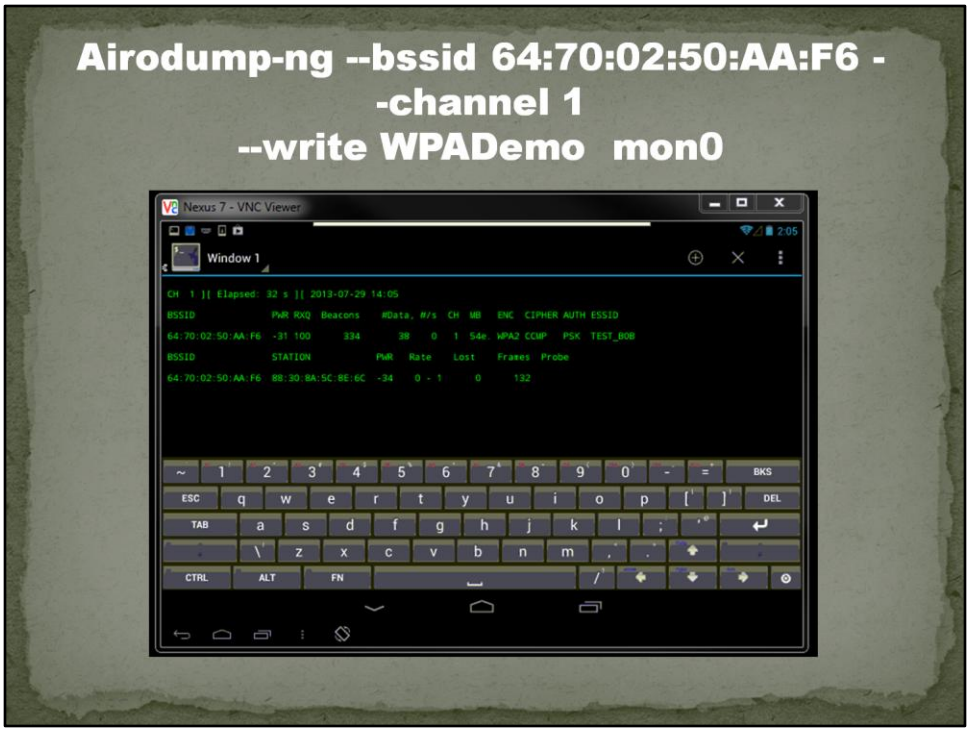


So armed with that info – how does WPA cracking work. You can see the diagram on the slide. It will keep trying words until the MIC matches.

I put the Oxford English dictionary here for a reason. Like any dictionary attack this one is only as good as the quality of the dictionary you are using. If you were targeting a network in Germany a german dictionary would make more sense, or you could take a list of common passwords.

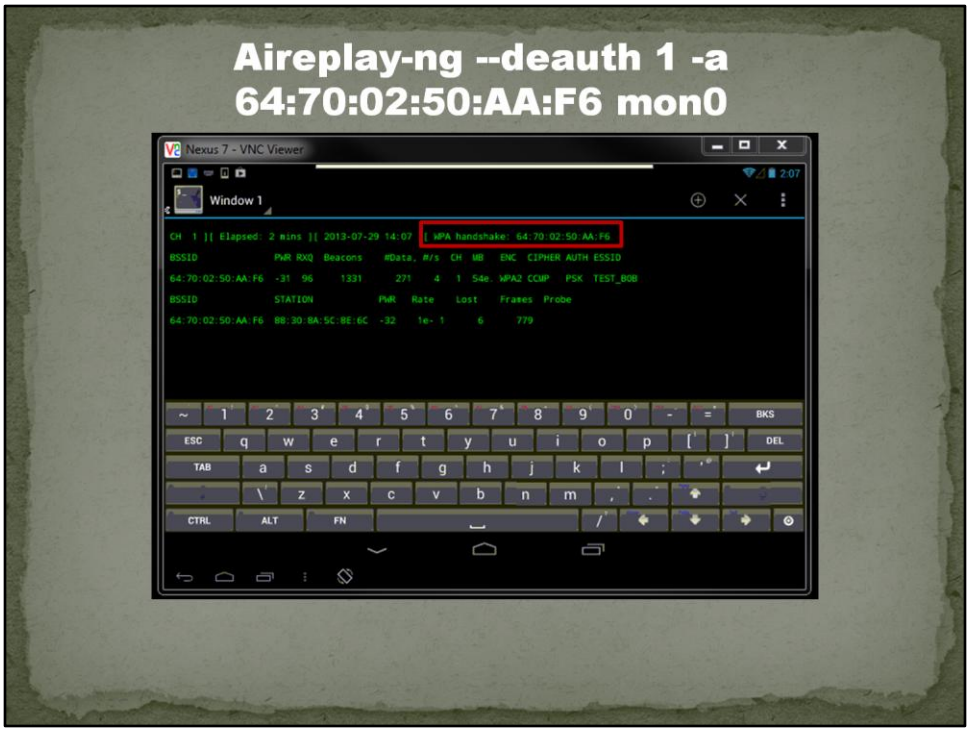
For this demo I just downloaded a list of the 10000 most common passwords from http://xato.net/passwords/more-top-worst-passwords/ . We'll use that in our testing, but in reality you'll probably want a bigger dictionary for active WPA attacks. I have some links to dictionaries on the slide – but be warned they can get quite big (several GBs). This combined with the fact that everyone has their own favourite dictionary are the reasons that the PWNPad ships with none.

Dictionaries:
http://maurisdump.blogspot.ie/2011/12/best-dictionaries-wordlist-for-wpa.html
http://wifi0wn.wordpress.com/wepwpawpa2-cracking-dictionary/



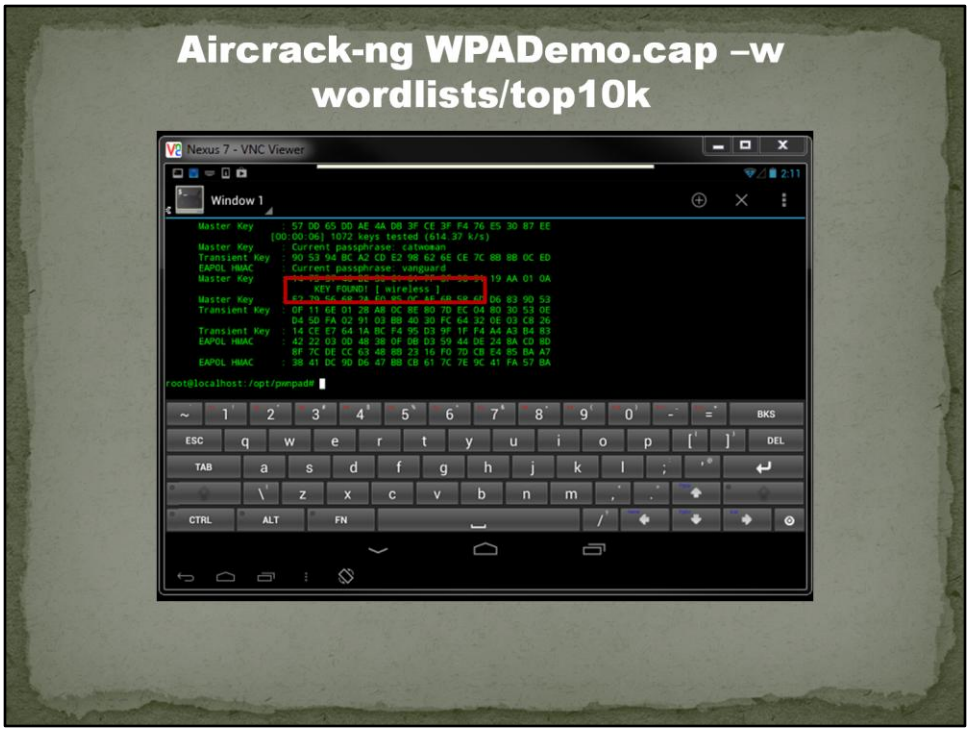
This time around we will start by doing the attack the manual way and then move onto doing it using Wifite. Again I'll step through it in slides and then do it live

First run airodump as shown so that we are saving the captured packets to a file.



Then either connect the client for the first time and capture the response or use a DeAuth packet to force it to reconnect

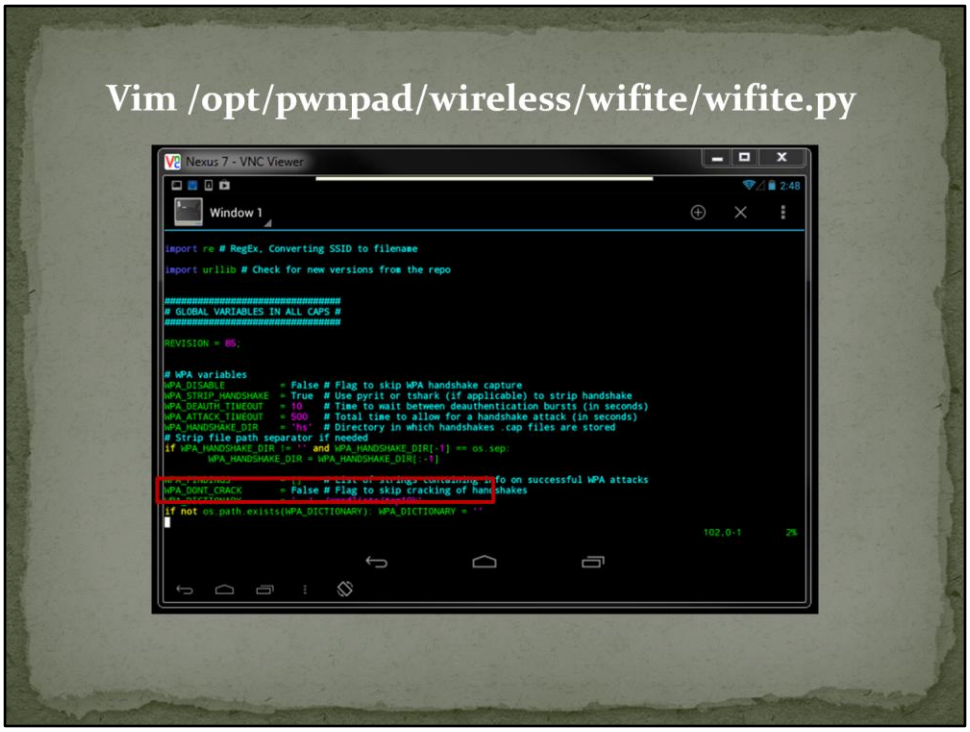
You'll know you have the handshake when the top right corner of Airodump shows WPA Handshake followed by the AP BSSID. Once thats done you can stop airodump.



Now simply run aircrack with your dictionary as a parameter, and after a short time it should complete and find the key!

On my PWNPad I was getting a speed of around 600 k/s . At that rate you could go through the current Oxford Dictionary in less than 5 mins ☺

Now that we've done it the hard way, lets try doing it with Wifite. There is one small problem however, wifite does not know where to find your dictionary.

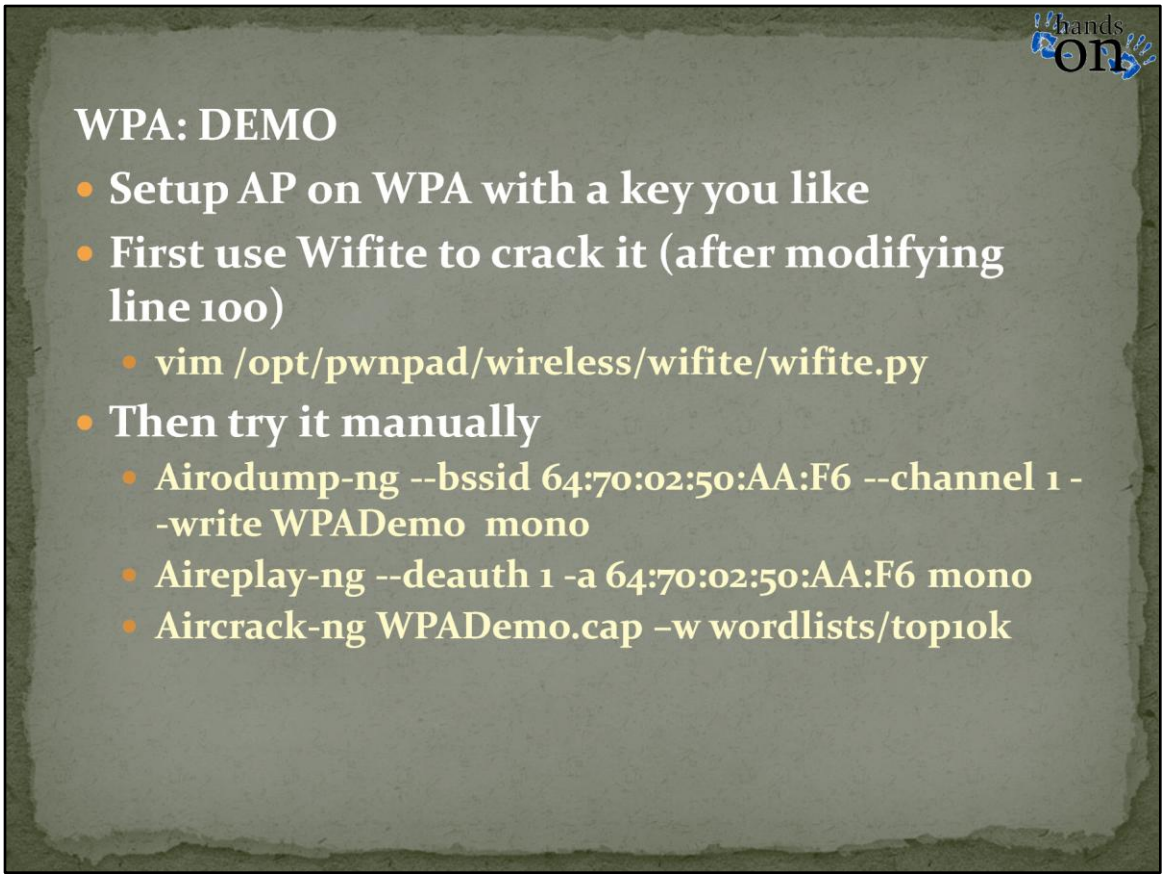


Open wifite in vim (or other editor) and go to line 100. Modify this to the path of your dictionary

(i for insert mode, esc to get back out, :w to write, :q to quit)

Then run Wifite and simply follow the steps.

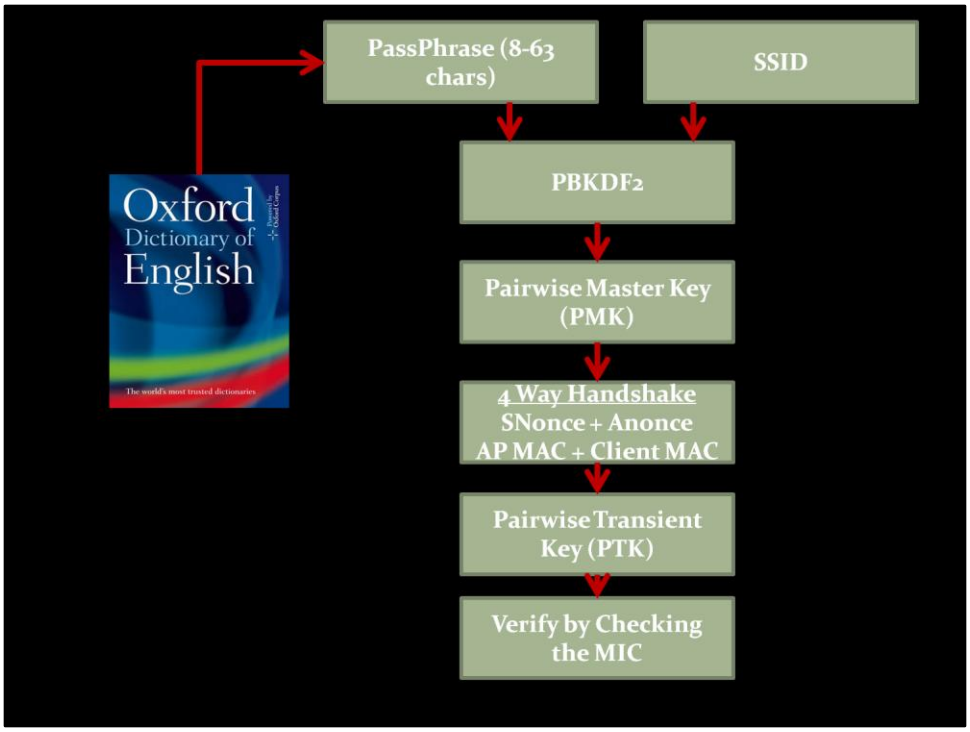
Note that Wifite actually carries out a lot more attacks that just the most common ones we have show in our slides. Rather than go through all of the various side attacks such as chop-chop, WPS etc – just be aware that they exist, and that Wifite will take care of them for you – but it is setup to always try the most successful attack types first.



DEMO TIME

Now try these yourself. Replace with your MAC addresses obviously ☺

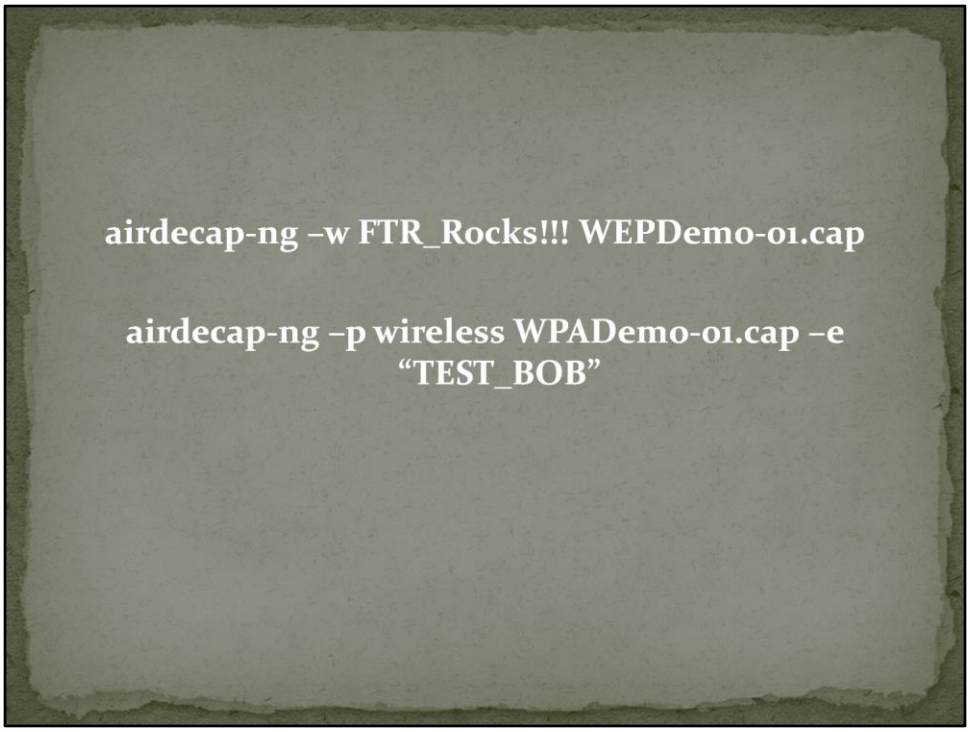
Congratulations – you now know how to crack the two most common Wifi encryptions!



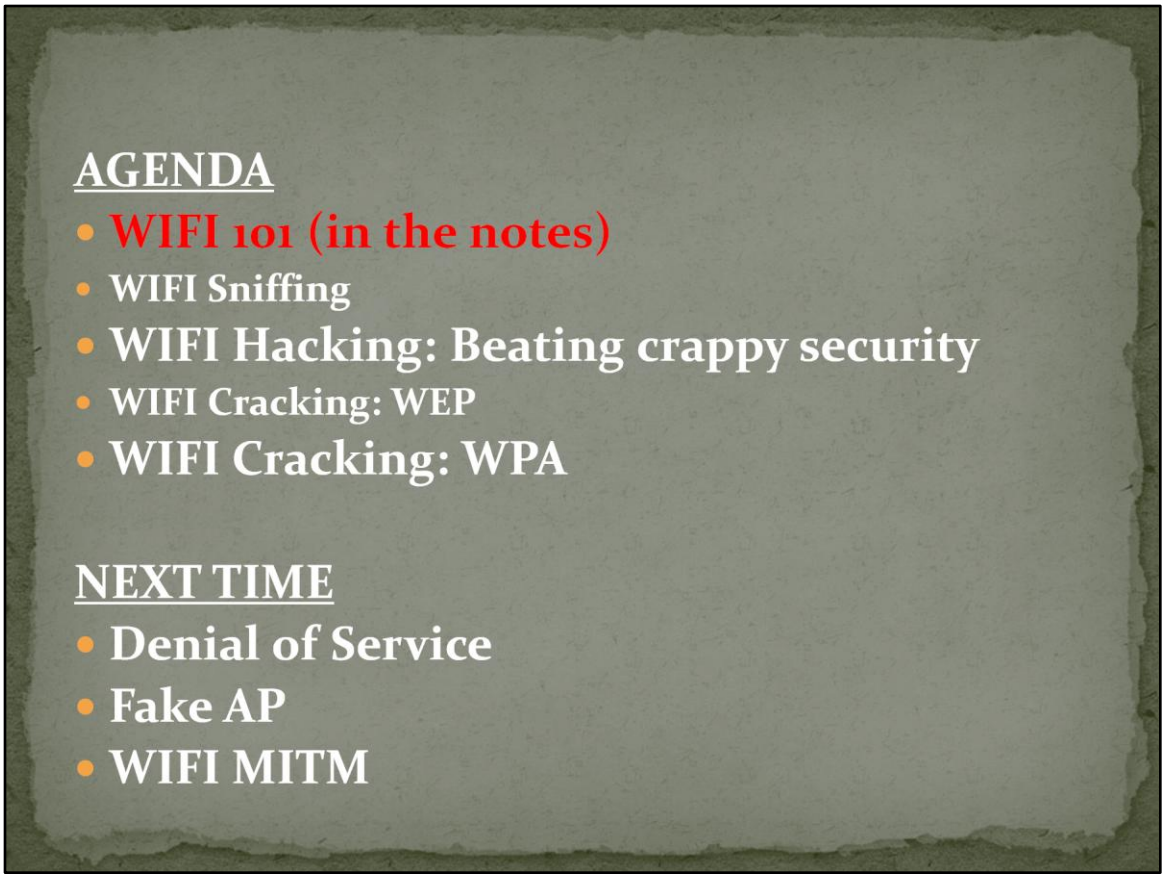
Before going onto the next part I wanted to bring this slide up again. The slowest part of the whole cracking of WPA is the generations of the PMKs . This is because the PBKDF2 algorithm actually performs over 4096 hashing functions to create the key. By comparison the calculation of the PTK from the other parameters is very computationally inexpensive.

So it would be nice to be able to build a rainbow table of passwords to PMKs in advance. However you'll notice that the PMK is also dependant on the SSID. The password "FTRRocks!" will generate a different key on a SSID "CorkSec" than it will on one called "MyHome".

There is a tool specifically for generating and using these Rainbow tables, but it is not included in PWNPad. Its called CowPatty and you can find it on Kali. You can also check torrent sites and download pre-created rainbow tables for the most common SSIDs such as linksys, McDonalds etc



Also for the sake of completeness. If you ever want to decrypt a PCAP once you have the password you can do using airdecap. You can see syntax for WEP and WPA on the slide.



So thats it for Wifi hacking 101 – you now know how to gain access to both WEP and WPA networks

I'll run a follow up Wifi session at CorkSec at a future meeting – because there is still lots of fun things to look at such as Denial of Service attacks, Fake Access points, and of the course the really powerful MITM attacks.

Until then- that is for now. Enjoy the rest of the night!