# Proxmark3 Handbook

**RFID || NFC reading, writing, cracking, and simulating**

**By Ahmed Alroky**

## Table of contents:

# Introduction:

In this book I will tell you a bit about RFID, NFC and how to attack these technologies, and will focus on the most common and the most well-known tool in this field, and it calls Proxmark3, by using it you will be able to read, clone and simulate both high and low frequency tags and do more interesting stuff such as cracking RFID UIDs and sniff cards, I hope you will find it useful and if you have any recommendations, please give me your feedback.

**Ahmed Alroky**

## What is RFID

RFID (radio frequency identification) is a form of wireless communication that incorporates the use of electromagnetic or electrostatic coupling in the radio frequency portion of the electromagnetic spectrum to uniquely identify an object, animal, or person.

## Different types of RFID

- **Low-frequency RFID systems.**

These range from 30 KHZ to 500 KHZ, though the typical frequency is 125 KHz. LF RFID has short transmission ranges, generally anywhere from a few inches to less than six feet.

- **High-frequency RFID system**

These range from 3 MHz to 30 MHz, with the typical HF frequency being 13.56 MHZ. The standard range is anywhere from a few inches to several feet.

- **UHF RFID systems.**

These range from 300 MHz to 960 MHz, with the typical frequency of 433 MHz and can generally be read from 25-plus feet away.

- **Microwave RFID systems.**

These run at 2.45 GHZ and can be read from 30-plus feet away.

## RFID hacking tools

The most well-known RFID hacking tools is proxmark3 but there is another tool:

**Keysy:**

Is a new product that can backup up to four RFID access credentials into a small key fob form factor? It will consolidate them all on your keychain so you can leave the originals at home and avoid having to pay costly replacement fees should you lose one.

Source: hak5 website

**ICopy-X:**

The updated version of proxmark3 tool, supports reading, simulating RFID\NFC tags and more functions.

**Chameleon:**

Chameleon tiny (or Mini) can read, write, and simulate NFC tags only, what so special about chameleon tiny is the small size you can bring it with you any place in your keychain

## RFID vs NFC

| | RFID | NFC |
|---|---|---|
| **Communication** | Unidirectional | Bidirectional |
| **Range** | Up to 100M | Less than 0.2M |
| **Bitrate** | Varies with frequency | Up to 424 kbit/s |
| **Continuous sampling** | No | Yes |

# Getting started with proxmark3

## different versions

Proxmark3 has many versions including: proxmark3, proxmark3 rdv2, proxmark3 easy, proxmark3 evo, proxmark3 rdv4 and ICopy-x version, but I will focus on one that o already owns while writing this guide "proxmark3 easy"

## Different versions of firmware

There are a lot of firmware forks across GitHub, but I will mention two of it:

- **The official Proxmark3 version:**

  Unfortunately, out of date and I rarely use it

- **Iceman Fork version:**

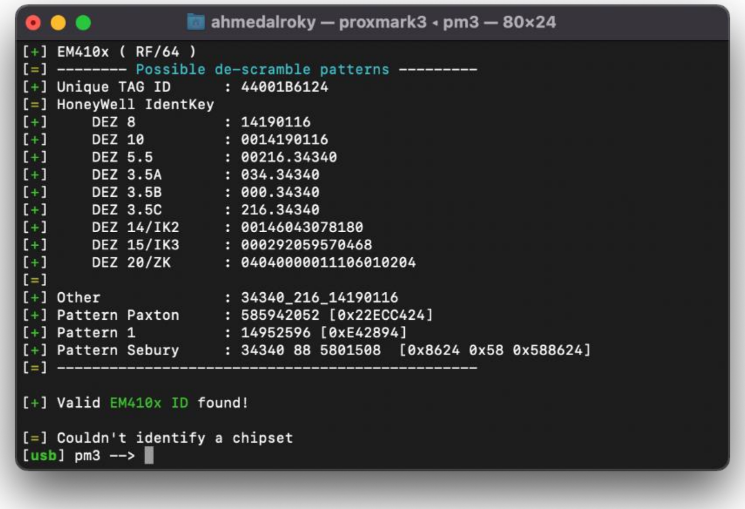
I use this one since I've found a lot of problems in the official version, and my advice to every Proxmark3 owner to use it because of the huge options and compatibility it come with including the easiest way to use proxmark3 via standalone mode as I will mention later.

## Compiling

Simply browse to https://github.com/RfidResearchGroup/proxmark3 and follow the instruction based on your host operating system and after compiling you can verify it's working by typing **pm3** in your terminal and press enter it should ask you to connect your proxmark3 device.

## Identify tag frequency and tag types

**lf search**



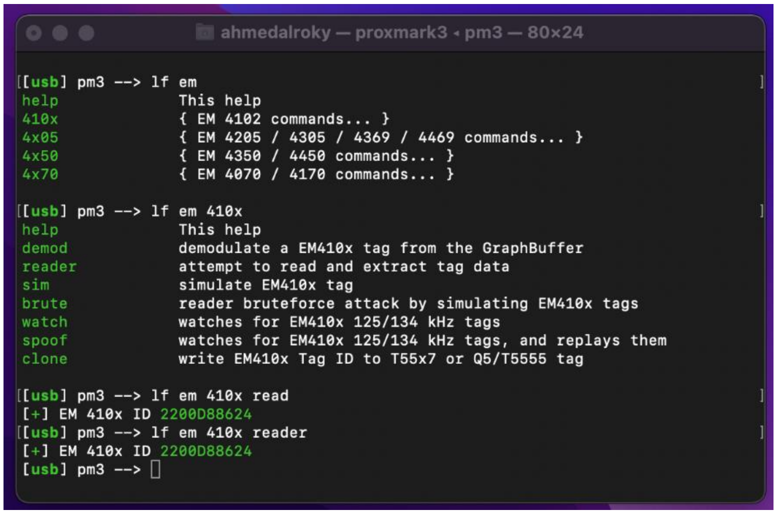
```
[+] EM410x ( RF/64 )
[=] -------- Possible de-scramble patterns ---------
[+] Unique TAG ID      : 44001B6124
[=] HoneyWell IdentKey
[+]     DEZ 8          : 14190116
[+]     DEZ 10         : 0014190116
[+]     DEZ 5.5        : 00216.34340
[+]     DEZ 3.5A       : 034.34340
[+]     DEZ 3.5B       : 000.34340
[+]     DEZ 3.5C       : 216.34340
[+]     DEZ 14/IK2     : 00146043078180
[+]     DEZ 15/IK3     : 000292059570468
[+]     DEZ 20/ZK      : 04040000011106010204
[=]
[+] Other              : 34340_216_14190116
[+] Pattern Paxton     : 585942052 [0x22ECC424]
[+] Pattern 1          : 14952596 [0xE42894]
[+] Pattern Sebury     : 34340 88 5801508  [0x8624 0x58 0x588624]
[=] ------------------------------------------------

[+] Valid EM410x ID found!

[=] Couldn't identify a chipset
[usb] pm3 --> 
```

## Low frequency tags

### Reading a tag

** I will use the same previous EM4100 tag**

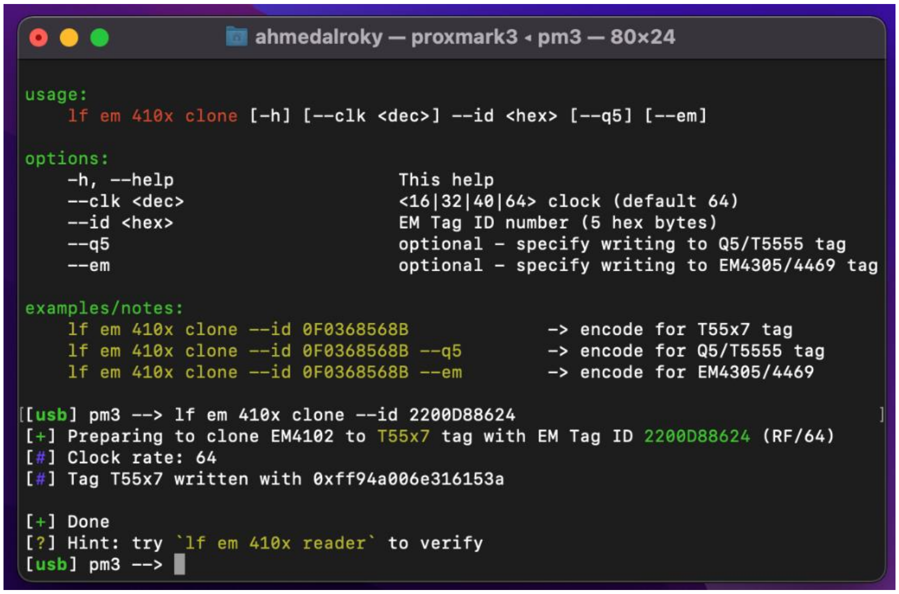
**lf em 410x reader**



```
[[usb] pm3 --> lf em                                                           ]
help                 This help
410x                 { EM 4102 commands... }
4x05                 { EM 4205 / 4305 / 4369 / 4469 commands... }
4x50                 { EM 4350 / 4450 commands... }
4x70                 { EM 4070 / 4170 commands... }

[[usb] pm3 --> lf em 410x                                                      ]
help                 This help
demod                demodulate a EM410x tag from the GraphBuffer
reader               attempt to read and extract tag data
sim                  simulate EM410x tag
brute                reader bruteforce attack by simulating EM410x tags
watch                watches for EM410x 125/134 kHz tags
spoof                watches for EM410x 125/134 kHz tags, and replays them
clone                write EM410x Tag ID to T55x7 or Q5/T5555 tag

[[usb] pm3 --> lf em 410x read                                                 ]
[+] EM 410x ID 2200D88624
[[usb] pm3 --> lf em 410x reader                                               ]
[+] EM 410x ID 2200D88624
[usb] pm3 --> 
```

## Cloning a tag

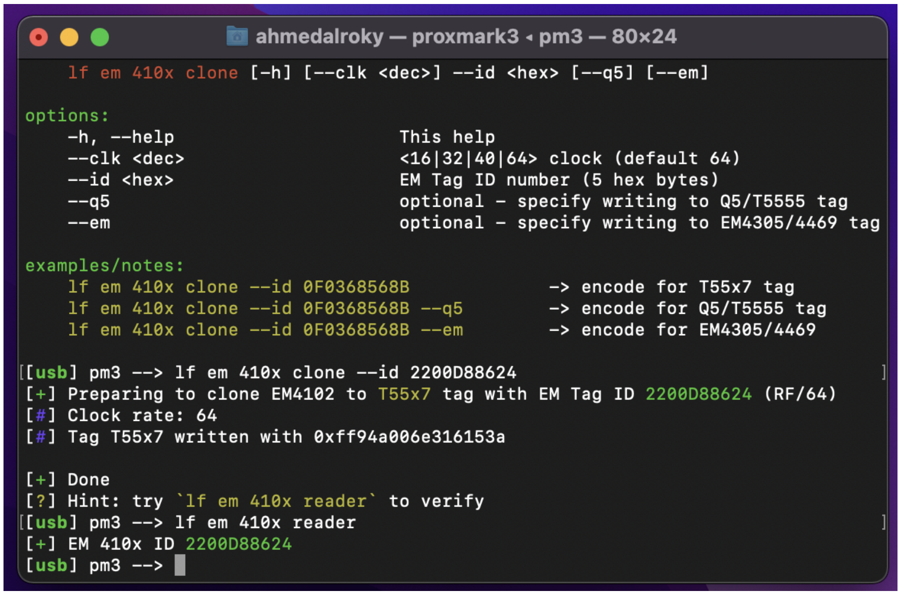
**lf em 410x clone –id <Tag UID>**



```
usage:
    lf em 410x clone [-h] [--clk <dec>] --id <hex> [--q5] [--em]

options:
    -h, --help                     This help
    --clk <dec>                    <16|32|40|64> clock (default 64)
    --id <hex>                     EM Tag ID number (5 hex bytes)
    --q5                           optional - specify writing to Q5/T5555 tag
    --em                           optional - specify writing to EM4305/4469 tag

examples/notes:
    lf em 410x clone --id 0F0368568B              -> encode for T55x7 tag
    lf em 410x clone --id 0F0368568B --q5         -> encode for Q5/T5555 tag
    lf em 410x clone --id 0F0368568B --em         -> encode for EM4305/4469

[[usb] pm3 --> lf em 410x clone --id 2200D88624                                ]
[+] Preparing to clone EM4102 to T55x7 tag with EM Tag ID 2200D88624 (RF/64)
[#] Clock rate: 64
[#] Tag T55x7 written with 0xff94a006e316153a

[+] Done
[?] Hint: try `lf em 410x reader` to verify
[usb] pm3 -->
```

Verify cloned card

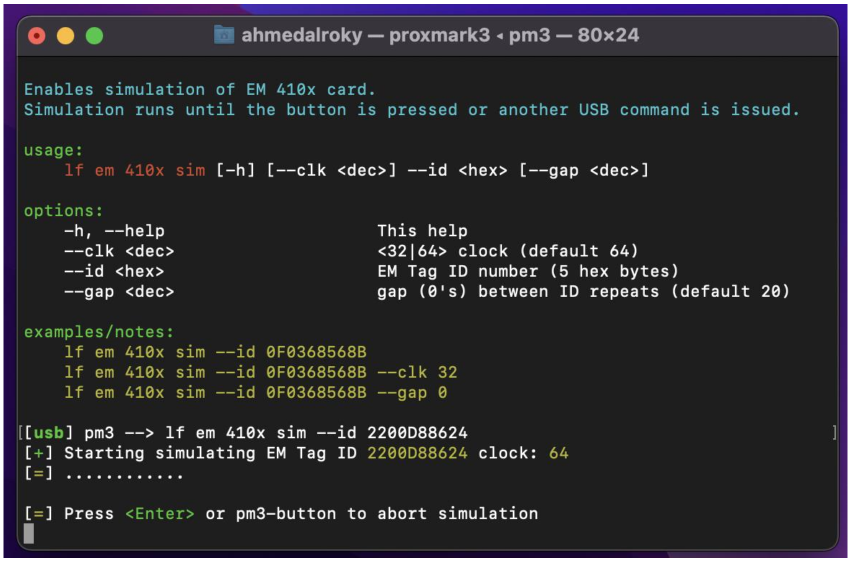
**lf search**



```
    lf em 410x clone [-h] [--clk <dec>] --id <hex> [--q5] [--em]

options:
    -h, --help                     This help
    --clk <dec>                    <16|32|40|64> clock (default 64)
    --id <hex>                     EM Tag ID number (5 hex bytes)
    --q5                           optional - specify writing to Q5/T5555 tag
    --em                           optional - specify writing to EM4305/4469 tag

examples/notes:
    lf em 410x clone --id 0F0368568B              -> encode for T55x7 tag
    lf em 410x clone --id 0F0368568B --q5         -> encode for Q5/T5555 tag
    lf em 410x clone --id 0F0368568B --em         -> encode for EM4305/4469

[[usb] pm3 --> lf em 410x clone --id 2200D88624                                ]
[+] Preparing to clone EM4102 to T55x7 tag with EM Tag ID 2200D88624 (RF/64)
[#] Clock rate: 64
[#] Tag T55x7 written with 0xff94a006e316153a

[+] Done
[?] Hint: try `lf em 410x reader` to verify
[[usb] pm3 --> lf em 410x reader                                               ]
[+] EM 410x ID 2200D88624
[usb] pm3 -->
```

## Simulating a tag

**lf em em410x sim –id <Tag UID>**

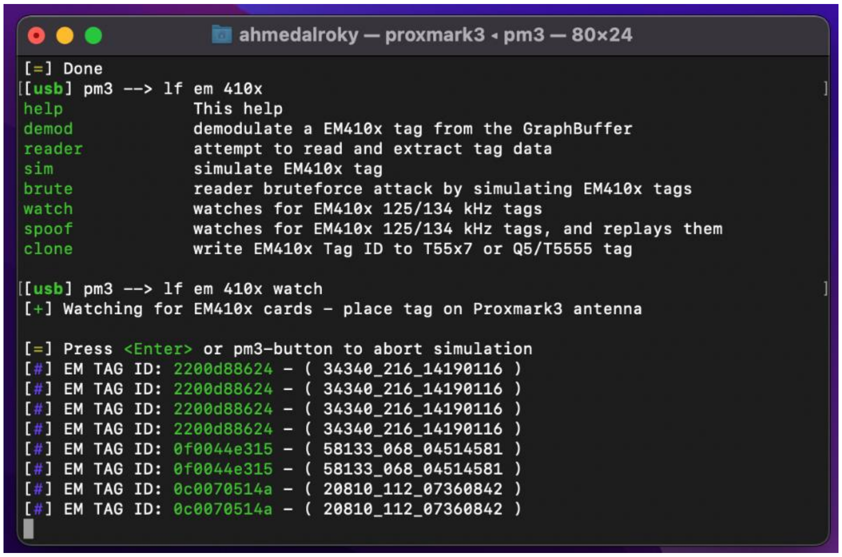


```
Enables simulation of EM 410x card.
Simulation runs until the button is pressed or another USB command is issued.

usage:
    lf em 410x sim [-h] [--clk <dec>] --id <hex> [--gap <dec>]

options:
    -h, --help                     This help
    --clk <dec>                    <32|64> clock (default 64)
    --id <hex>                     EM Tag ID number (5 hex bytes)
    --gap <dec>                    gap (0's) between ID repeats (default 20)

examples/notes:
    lf em 410x sim --id 0F0368568B
    lf em 410x sim --id 0F0368568B --clk 32
    lf em 410x sim --id 0F0368568B --gap 0

[[usb] pm3 --> lf em 410x sim --id 2200D88624                                  ]
[+] Starting simulating EM Tag ID 2200D88624 clock: 64
[=] ............

[=] Press <Enter> or pm3-button to abort simulation
```

## Watching for tags

** This command will launch low frequency reader continuously until you stop it **

**lf em em410x watch**

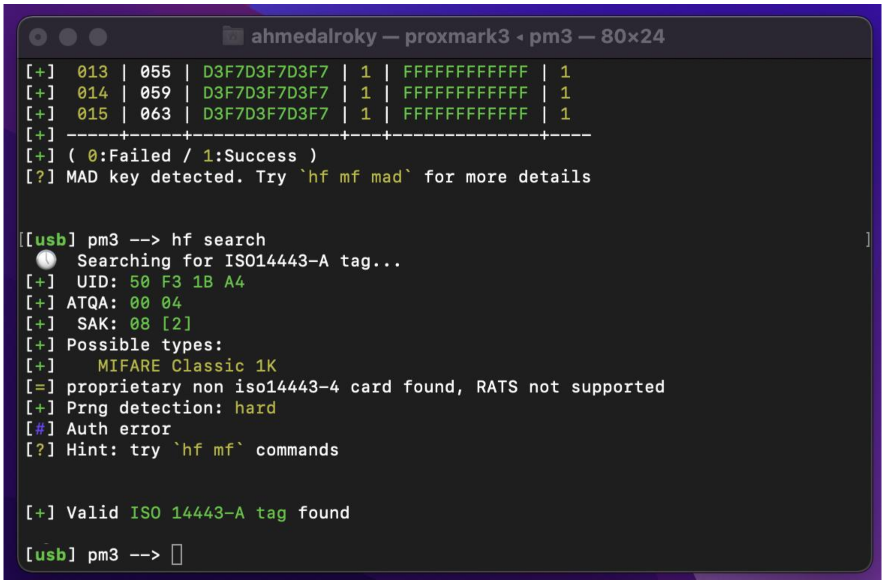


```
[=] Done
[[usb] pm3 --> lf em 410x                                                      ]
help                 This help
demod                demodulate a EM410x tag from the GraphBuffer
reader               attempt to read and extract tag data
sim                  simulate EM410x tag
brute                reader bruteforce attack by simulating EM410x tags
watch                watches for EM410x 125/134 kHz tags
spoof                watches for EM410x 125/134 kHz tags, and replays them
clone                write EM410x Tag ID to T55x7 or Q5/T5555 tag

[[usb] pm3 --> lf em 410x watch                                                ]
[+] Watching for EM410x cards - place tag on Proxmark3 antenna

[=] Press <Enter> or pm3-button to abort simulation
[#] EM TAG ID: 2200d88624 - ( 34340_216_14190116 )
[#] EM TAG ID: 2200d88624 - ( 34340_216_14190116 )
[#] EM TAG ID: 2200d88624 - ( 34340_216_14190116 )
[#] EM TAG ID: 2200d88624 - ( 34340_216_14190116 )
[#] EM TAG ID: 0f0044e315 - ( 58133_068_04514581 )
[#] EM TAG ID: 0f0044e315 - ( 58133_068_04514581 )
[#] EM TAG ID: 0c0070514a - ( 20810_112_07360842 )
[#] EM TAG ID: 0c0070514a - ( 20810_112_07360842 )
```

# high frequency tags

## identifying a tag

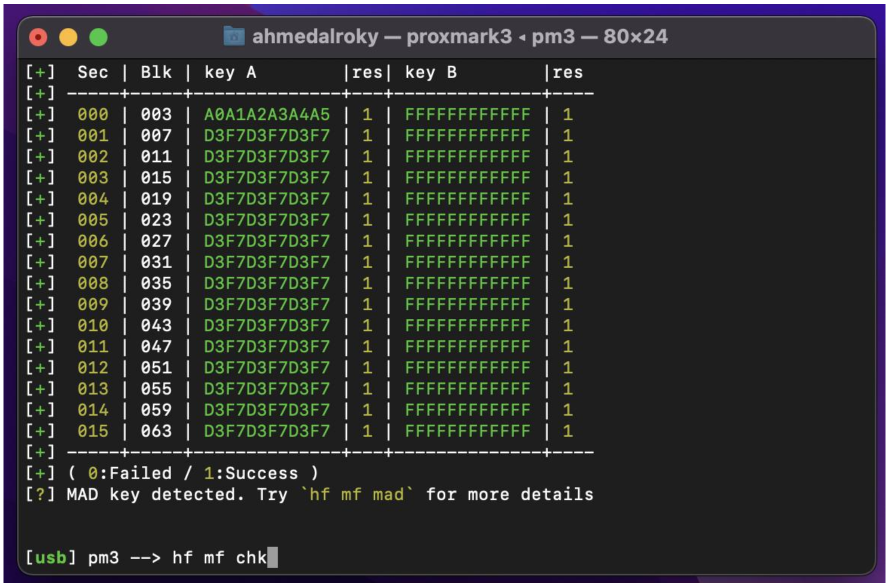
**hf search**



```
[+]  013 | 055 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  014 | 059 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  015 | 063 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+] -----+-----+--------------+---+--------------+----
[+] ( 0:Failed / 1:Success )
[?] MAD key detected. Try `hf mf mad` for more details


[usb] pm3 --> hf search
    Searching for ISO14443-A tag...
[+]  UID: 50 F3 1B A4
[+] ATQA: 00 04
[+]  SAK: 08 [2]
[+] Possible types:
[+]    MIFARE Classic 1K
[=] proprietary non iso14443-4 card found, RATS not supported
[+] Prng detection: hard
[#] Auth error
[?] Hint: try `hf mf` commands


[+] Valid ISO 14443-A tag found

[usb] pm3 --> 
```

## Cracking 1K Mifare tags

**hf mf chk**

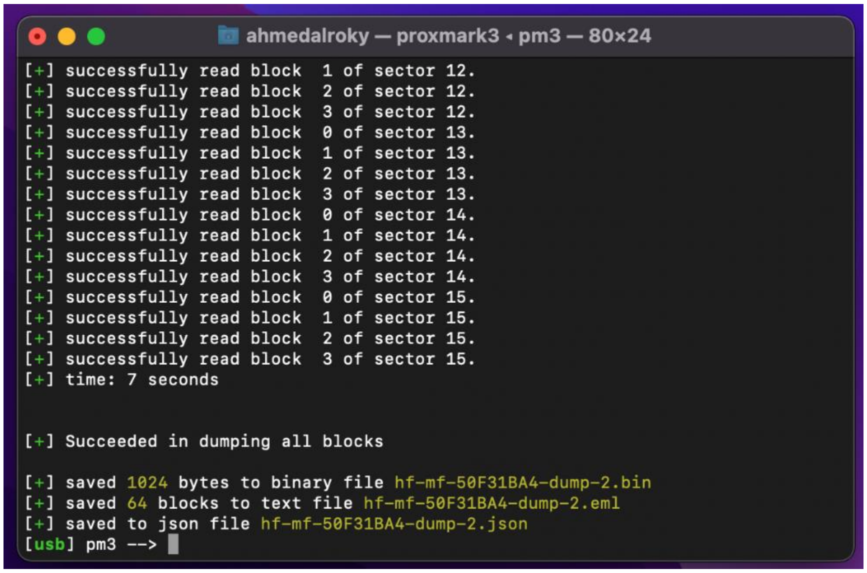


```
[+]  Sec | Blk | key A        |res| key B        |res
[+] -----+-----+--------------+---+--------------+----
[+]  000 | 003 | A0A1A2A3A4A5 | 1 | FFFFFFFFFFFF | 1
[+]  001 | 007 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  002 | 011 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  003 | 015 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  004 | 019 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  005 | 023 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  006 | 027 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  007 | 031 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  008 | 035 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  009 | 039 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  010 | 043 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  011 | 047 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  012 | 051 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  013 | 055 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  014 | 059 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+]  015 | 063 | D3F7D3F7D3F7 | 1 | FFFFFFFFFFFF | 1
[+] -----+-----+--------------+---+--------------+----
[+] ( 0:Failed / 1:Success )
[?] MAD key detected. Try `hf mf mad` for more details


[usb] pm3 --> hf mf chk
```

## cloning tags

### dump from a card to a file

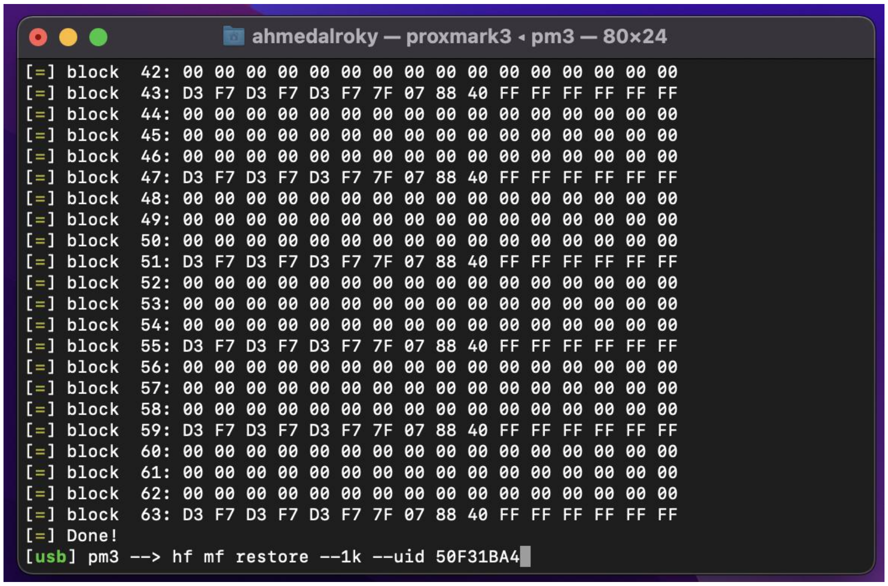
**hf mf dump**



```
[+] successfully read block  1 of sector 12.
[+] successfully read block  2 of sector 12.
[+] successfully read block  3 of sector 12.
[+] successfully read block  0 of sector 13.
[+] successfully read block  1 of sector 13.
[+] successfully read block  2 of sector 13.
[+] successfully read block  3 of sector 13.
[+] successfully read block  0 of sector 14.
[+] successfully read block  1 of sector 14.
[+] successfully read block  2 of sector 14.
[+] successfully read block  3 of sector 14.
[+] successfully read block  0 of sector 15.
[+] successfully read block  1 of sector 15.
[+] successfully read block  2 of sector 15.
[+] successfully read block  3 of sector 15.
[+] time: 7 seconds


[+] Succeeded in dumping all blocks

[+] saved 1024 bytes to binary file hf-mf-50F31BA4-dump-2.bin
[+] saved 64 blocks to text file hf-mf-50F31BA4-dump-2.eml
[+] saved to json file hf-mf-50F31BA4-dump-2.json
[usb] pm3 --> 
```

### Restore from file to a card

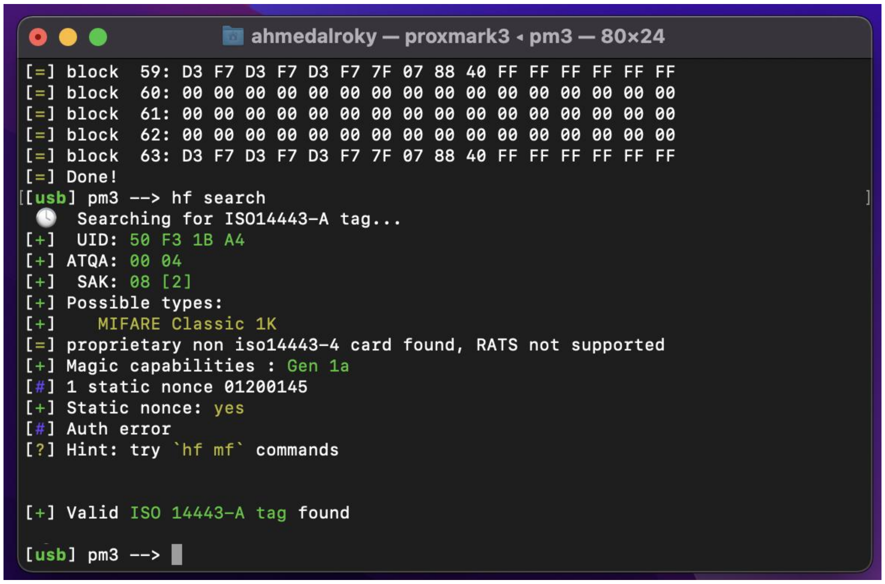
**hf mf restore**



```
[=] block  42: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  43: D3 F7 D3 F7 D3 F7 7F 07 88 40 FF FF FF FF FF FF
[=] block  44: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  45: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  46: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  47: D3 F7 D3 F7 D3 F7 7F 07 88 40 FF FF FF FF FF FF
[=] block  48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  49: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  51: D3 F7 D3 F7 D3 F7 7F 07 88 40 FF FF FF FF FF FF
[=] block  52: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  53: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  54: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  55: D3 F7 D3 F7 D3 F7 7F 07 88 40 FF FF FF FF FF FF
[=] block  56: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  57: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  58: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  59: D3 F7 D3 F7 D3 F7 7F 07 88 40 FF FF FF FF FF FF
[=] block  60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  61: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  62: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  63: D3 F7 D3 F7 D3 F7 7F 07 88 40 FF FF FF FF FF FF
[=] Done!
[usb] pm3 --> hf mf restore --1k --uid 50F31BA4
```

## confirm cloned card

**hf search**

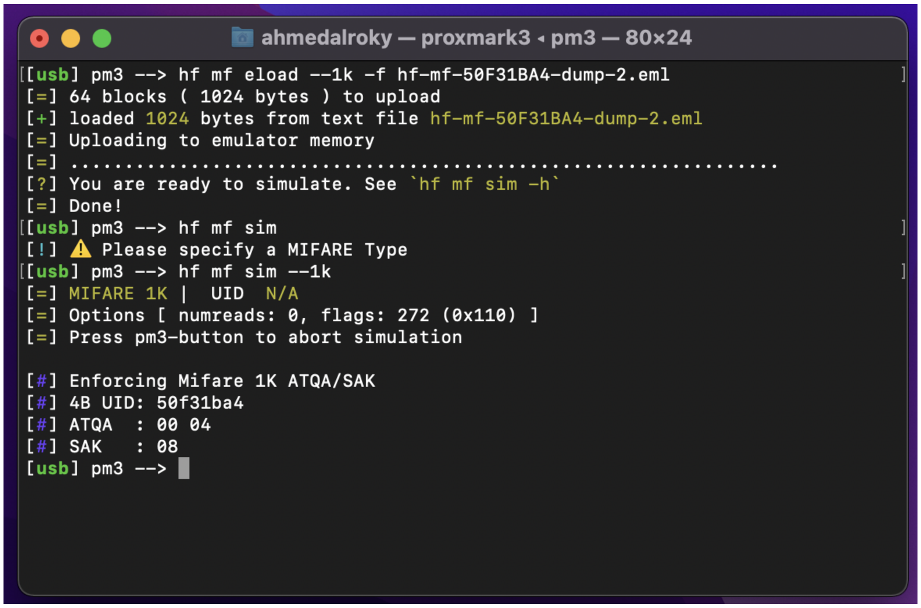


```
[=] block  59: D3 F7 D3 F7 D3 F7 7F 07 88 40 FF FF FF FF FF FF
[=] block  60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  61: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  62: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block  63: D3 F7 D3 F7 D3 F7 7F 07 88 40 FF FF FF FF FF FF
[=] Done!
[usb] pm3 --> hf search
 🕓  Searching for ISO14443-A tag...
[+]  UID: 50 F3 1B A4
[+] ATQA: 00 04
[+]  SAK: 08 [2]
[+] Possible types:
[+]    MIFARE Classic 1K
[=] proprietary non iso14443-4 card found, RATS not supported
[+] Magic capabilities : Gen 1a
[#] 1 static nonce 01200145
[+] Static nonce: yes
[#] Auth error
[?] Hint: try `hf mf` commands


[+] Valid ISO 14443-A tag found

[usb] pm3 --> 
```

## Simulating tag

**hf mf eload –1k -f <Dump File Name>**

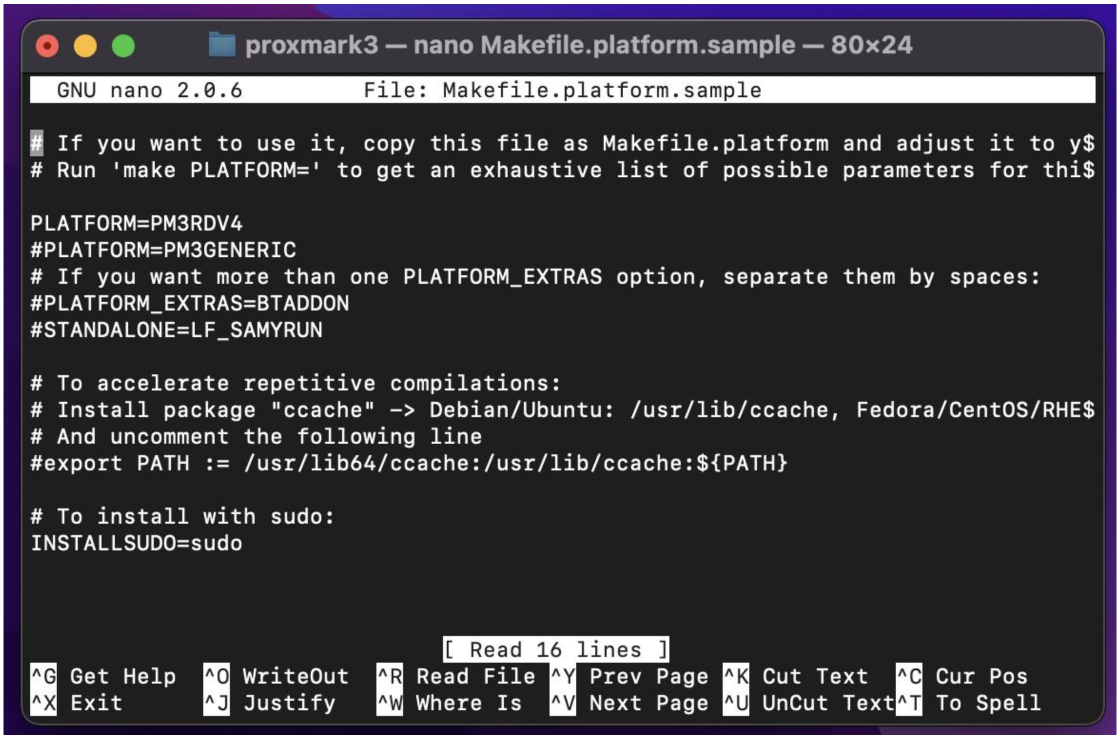
**hf mf sim –1k**



```
[usb] pm3 --> hf mf eload --1k -f hf-mf-50F31BA4-dump-2.eml
[=] 64 blocks ( 1024 bytes ) to upload
[+] loaded 1024 bytes from text file hf-mf-50F31BA4-dump-2.eml
[=] Uploading to emulator memory
[=] ................................................................
[?] You are ready to simulate. See `hf mf sim -h`
[=] Done!
[usb] pm3 --> hf mf sim
[!] ⚠ Please specify a MIFARE Type
[usb] pm3 --> hf mf sim --1k
[=] MIFARE 1K |  UID  N/A
[=] Options [ numreads: 0, flags: 272 (0x110) ]
[=] Press pm3-button to abort simulation

[#] Enforcing Mifare 1K ATQA/SAK
[#] 4B UID: 50f31ba4
[#] ATQA  : 00 04
[#] SAK   : 08
[usb] pm3 --> 
```

## proxmark3 standalone mode

To modify the functionalities of proxmark3’s standalone mode make a copy of “**Makefile.platform.sample**” to “**Makefile.platform**” and modify it based on your proxmark3 model (uncomment **PLATFORM=PM3GENERIC** and but a # before **PLATFORM=PM3RDV4** if you have a different version of proxmark3) or if you have **rdv4** version keep this lines , un comment STANDALONE and put your mode based on this WIKI
https://github.com/RfidResearchGroup/proxmark3/wiki/Standalone-mode



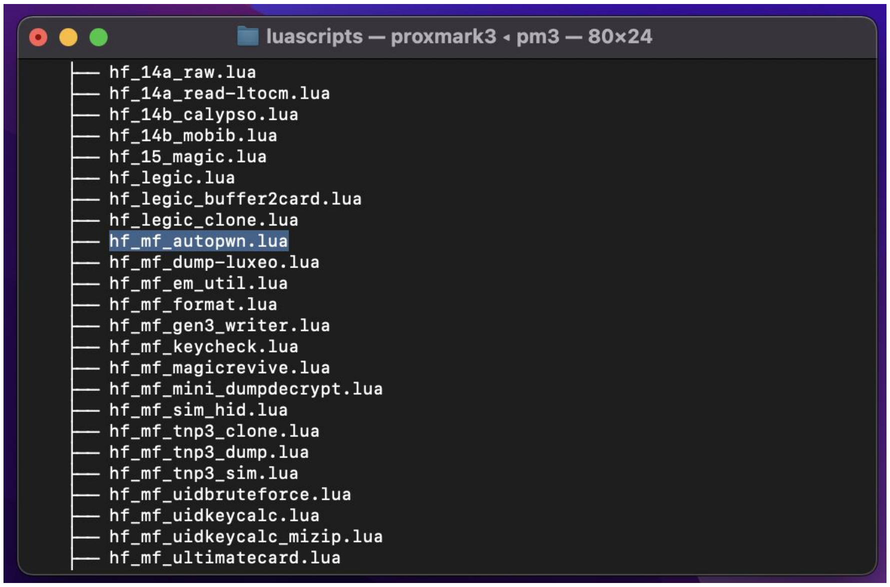
```
  GNU nano 2.0.6          File: Makefile.platform.sample

# If you want to use it, copy this file as Makefile.platform and adjust it to y$
# Run 'make PLATFORM=' to get an exhaustive list of possible parameters for thi$

PLATFORM=PM3RDV4
#PLATFORM=PM3GENERIC
# If you want more than one PLATFORM_EXTRAS option, separate them by spaces:
#PLATFORM_EXTRAS=BTADDON
#STANDALONE=LF_SAMYRUN

# To accelerate repetitive compilations:
# Install package "ccache" -> Debian/Ubuntu: /usr/lib/ccache, Fedora/CentOS/RHE$
# And uncomment the following line
#export PATH := /usr/lib64/ccache:/usr/lib/ccache:${PATH}

# To install with sudo:
INSTALLSUDO=sudo



                               [ Read 16 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

# Proxmark3 scripting and automation

## List available lua scripts

You will find some preloaded scripts with **ICE man** fork version view all available lua scripts by typing:

**script list**

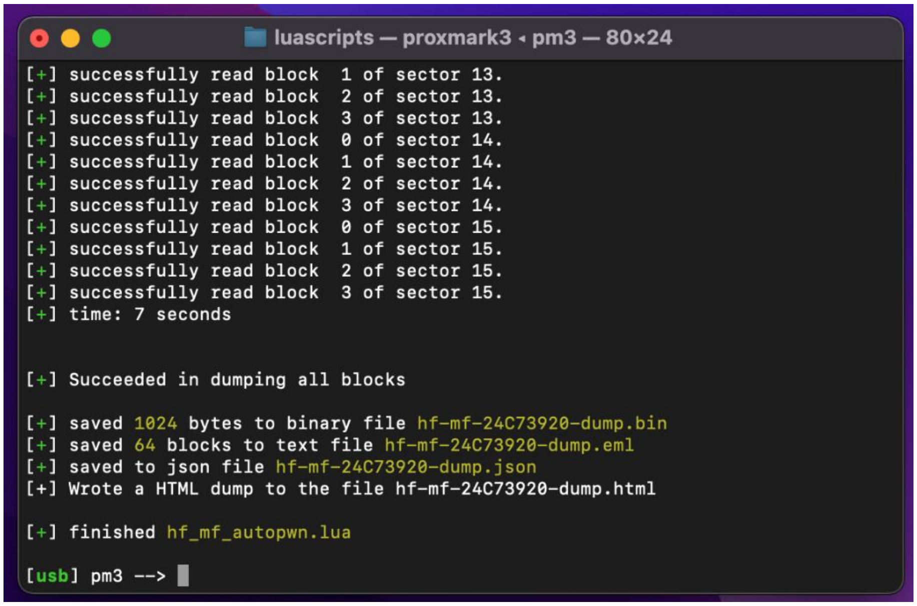


```
    ├── hf_14a_raw.lua
    ├── hf_14a_read-ltocm.lua
    ├── hf_14b_calypso.lua
    ├── hf_14b_mobib.lua
    ├── hf_15_magic.lua
    ├── hf_legic.lua
    ├── hf_legic_buffer2card.lua
    ├── hf_legic_clone.lua
    ├── hf_mf_autopwn.lua
    ├── hf_mf_dump-luxeo.lua
    ├── hf_mf_em_util.lua
    ├── hf_mf_format.lua
    ├── hf_mf_gen3_writer.lua
    ├── hf_mf_keycheck.lua
    ├── hf_mf_magicrevive.lua
    ├── hf_mf_mini_dumpdecrypt.lua
    ├── hf_mf_sim_hid.lua
    ├── hf_mf_tnp3_clone.lua
    ├── hf_mf_tnp3_dump.lua
    ├── hf_mf_tnp3_sim.lua
    ├── hf_mf_uidbruteforce.lua
    ├── hf_mf_uidkeycalc.lua
    ├── hf_mf_uidkeycalc_mizip.lua
    ├── hf_mf_ultimatecard.lua
```

## Run LUA script

And run your chosen script by typing:

**script run <Script Name>**



```
[+] successfully read block  1 of sector 13.
[+] successfully read block  2 of sector 13.
[+] successfully read block  3 of sector 13.
[+] successfully read block  0 of sector 14.
[+] successfully read block  1 of sector 14.
[+] successfully read block  2 of sector 14.
[+] successfully read block  3 of sector 14.
[+] successfully read block  0 of sector 15.
[+] successfully read block  1 of sector 15.
[+] successfully read block  2 of sector 15.
[+] successfully read block  3 of sector 15.
[+] time: 7 seconds


[+] Succeeded in dumping all blocks

[+] saved 1024 bytes to binary file hf-mf-24C73920-dump.bin
[+] saved 64 blocks to text file hf-mf-24C73920-dump.eml
[+] saved to json file hf-mf-24C73920-dump.json
[+] Wrote a HTML dump to the file hf-mf-24C73920-dump.html

[+] finished hf_mf_autopwn.lua

[usb] pm3 --> 
```

## extra features:

- Wigand manipulation
- RFID sniffing
- Cracking RFID reader “simulate all possible UIDs”
- Play with Smart cards “RDV4 only”

## Other resources

ICE Man WIKI

Unbrick proxmark3

Cheat sheets

Supported Tags

Proxmark3 command dump

## Contact Info

Email: ahmedalroky@gmail.com

Alt Email: alroky@icloud.com

Linkedin: https://www.linkedin.com/in/ahmedalroky

Youtube: https://youtube.com/c/ahmedalroky