



NEAR FIELD COMMUNICATION

TECNOLOGIA E SICUREZZA

Andrea Draghetti



Linux Day 27 Ottobre 2018





\$ WHOAMI

- + Phishing Analysis and Contrast @ D3Lab
- + Team Member @ BackBox Linux



+ NEAR FIELD COMMUNICATION

La tecnologia NFC si è evoluta da una combinazione d'identificazione senza contatto o RFID (Radio Frequency Identification – Identificazione a Radio Frequenza) e altre tecnologie di connettività. Contrariamente ai più semplici dispositivi RFID, NFC permette una comunicazione bidirezionale: quando due apparecchi NFC (lo initiator e il target) vengono accostati entro un raggio di 4 cm, viene creata una rete peer-to-peer tra i due ed entrambi possono inviare e ricevere informazioni.

La tecnologia NFC opera alla frequenza di 13,56 MHz e può raggiungere una velocità di trasmissione massima di 424 kbit/s.

{Wikipedia}



+ NFC VS BLUETOOTH



Bluetooth

- Permette il trasferimento dati ad alta velocità
- Consente la comunicazione a medie distanze
- Richiede pairing iniziale

NFC

- Comunicazione istantanea
- È orientato al trasferimento dati one-shot
- Può essere utilizzato per effettuare un pairing Bluetooth



+ NFC VS QR CODE

QR Code

- Tecnologia a costo zero
- Solo read-only
- Non offre protezione asimmetrica

NFC

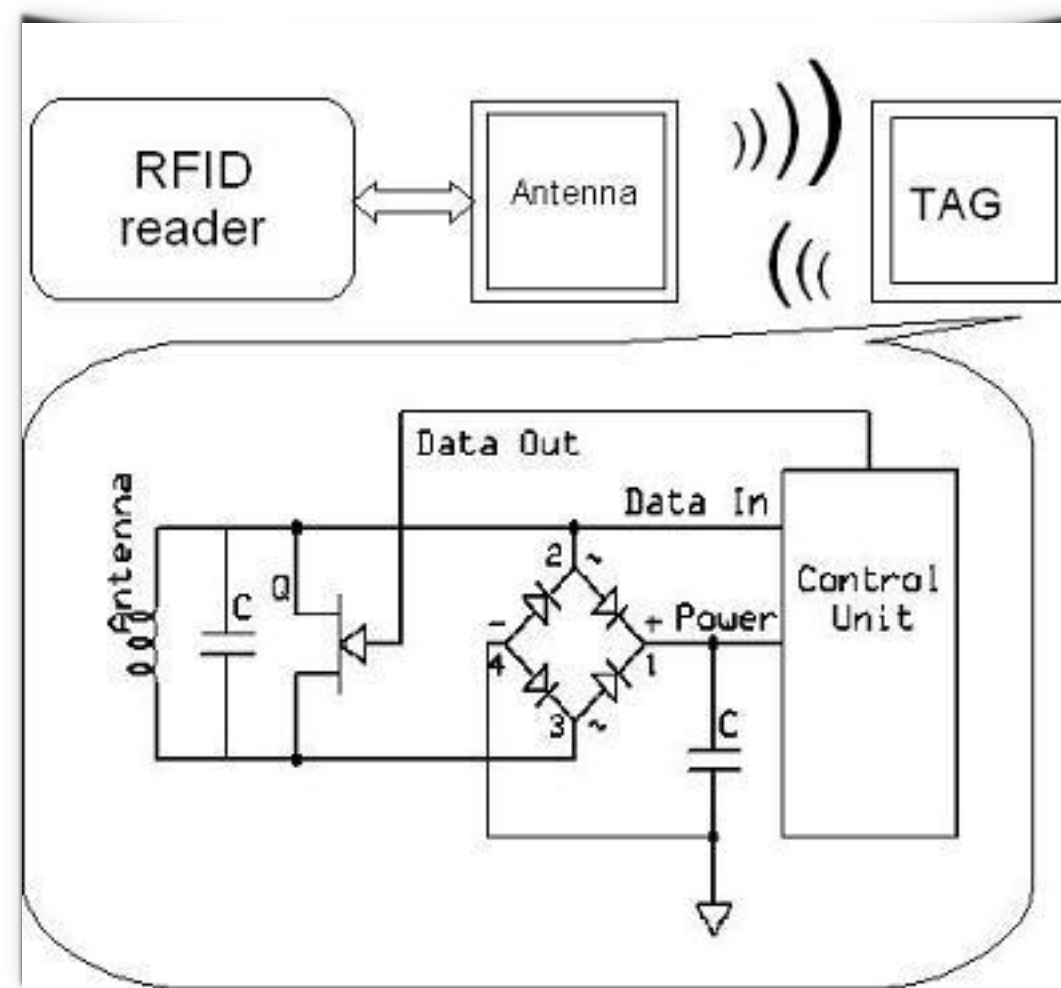
- Offre protezione simmetrica e asimmetrica
- I tag Nfc possono essere sia scritti che letti
- Richiede negli Smartphone la presenza del chipset



+ PRINCIPIO DI FUNZIONAMENTO

Il componente attivo emette un campo elettromagnetico che induce una corrente sul circuito del componente passivo.

Tale corrente è sufficiente ad alimentare il microcontrollore a bordo, ricevere il dato, effettuare l'elaborazione e trasmettere una risposta.



Fonte: <http://www.scienceprog.com/how-does-rfid-tag-technology-works/>



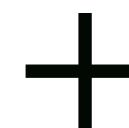
+ SCENARI APPLICATIVI



- Pagamenti
- Advertising
- Ticketing
- Tracking
- Gaming
- Social
- Medical



+ TAG NFC



+ CHIP NFC

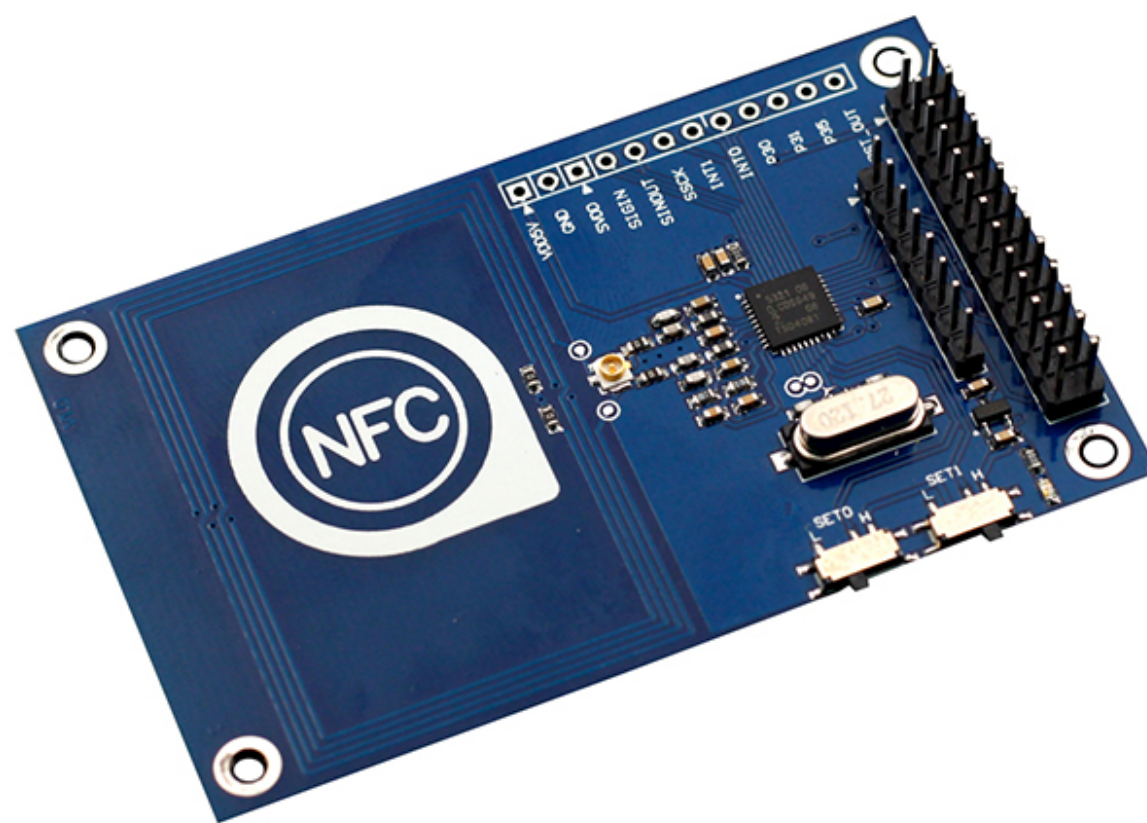
- **NTAG**: Chip di ultima generazione che garantiscono una piena compatibilità con gli smartphone (fino a 888byte)
- **Mifare Classic**: Supporto la crittografia e progettati per un prolungato utilizzo, raramente supportati dagli smartphone (fino a 3440byte)
- **Mifare Ultralight**: Come i precedenti ma con minor spazio di memoria (46byte)
- **iCode**: Standard industriale compatibile con gli standard ISO 15693 e ISO 18000-3



+ LETTORI NFC



ACR122U



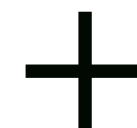
PN532



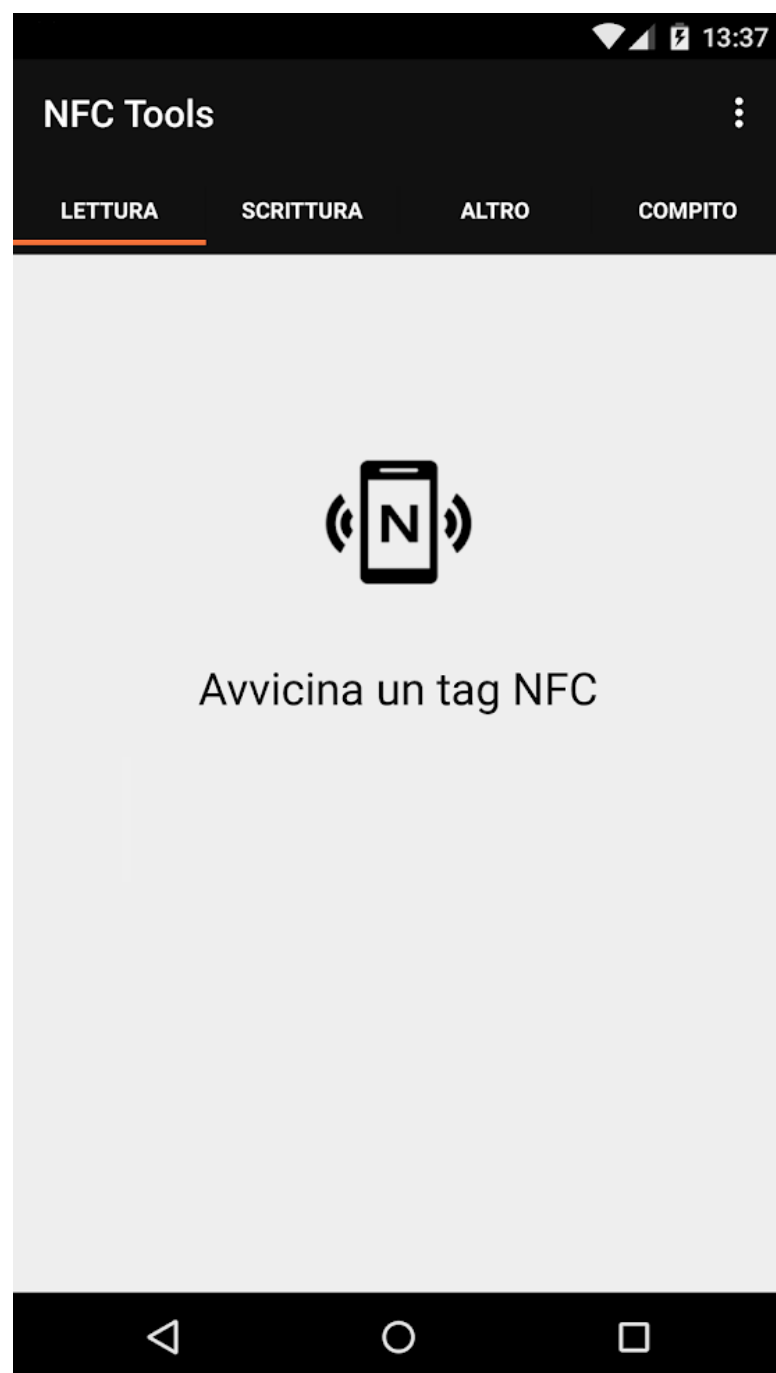
+ LETTORI NFC



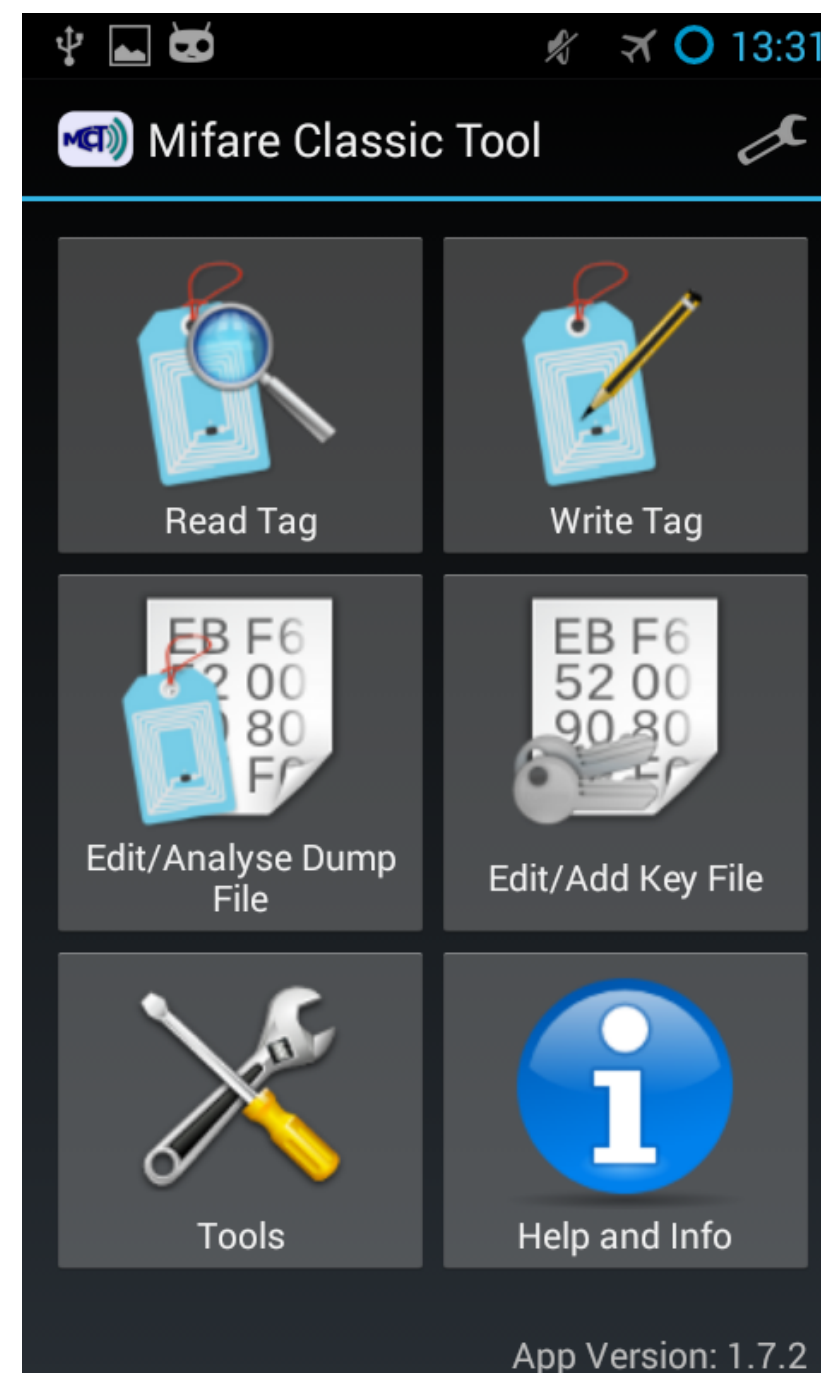
Proxmark3



+ APPLICAZIONI ANDROID



NFC Tools



MIFARE Classic Tool





APPLICAZIONI LINUX: MFOC

```
Terminale - drego85@backbox: ~
File Modifica Visualizza Terminale Schede Aiuto

drego85@backbox:~$ mfoc -h
Usage: mfoc [-h] [-k key]... [-P probnum] [-T tolerance] [-O output]

h    print this help and exit
k    try the specified key in addition to the default keys
P    number of probes per sector, instead of default of 20
T    nonce tolerance half-range, instead of default of 20
     (i.e., 40 for the total range, in both directions)
O    file in which the card contents will be written (REQUIRED)

Example: mfoc -O mycard.mfd
Example: mfoc -k ffffffffddd -O mycard.mfd
Example: mfoc -P 50 -T 30 -O mycard.mfd

This is mfoc version 0.10.2.
For more information, run: 'man mfoc'.
drego85@backbox:~$
```

<https://github.com/nfc-tools/mfoc>



+ APPLICAZIONI LINUX: MFCUK

```
Terminale - drego85@backbox: ~
File Modifica Visualizza Terminale Schede Aiuto
mfcuk - 0.3.2
Mifare Classic DarkSide Key Recovery Tool - 0.3
by Andrei Costin, zveriu@gmail.com, http://andreicostin.com

Usage:
-C - require explicit connection to the reader. Without this option, the connection is not made and recovery will not occur
-i mifare.dmp - load input mifare_classic_tag type dump
-I mifare_ext.dmp - load input extended dump specific to this tool, has several more fields on top of mifare_classic_tag type dump
-o mifare.dmp - output the resulting mifare_classic_tag dump to a given file
-O mifare_ext.dmp - output the resulting extended dump to a given file
-V sector[:A/B/any_other_alphanum[:fullkey]] - verify key for specified sector, -1 means all sectors
    After first semicolon key-type can specified: A verifies only keyA, B verifies only keyB, anything else verifies both keys
    After second semicolon full 12 hex-digits key can specified - this key will override any loaded dump key for the given sector(s) and key-type(s)
-R sector[:A/B/any_other_alphanum] - recover key for sector, -1 means all sectors.
    After first semicolon key-type can specified: A recovers only keyA, B recovers only keyB, anything else recovers both keys
-U UID - force specific UID. If a dump was loaded with -i, -U will overwrite the in the memory where dump was loaded
-M tagtype - force specific tagtype. 8 is 1K, 24 is 4K, 32 is DESFire
-D - for sectors and key-types marked for verification, in first place use default keys to verify (maybe you are lucky)
-d key - specifies additional full 12 hex-digits default key to be checked. Multiple -d options can be used for more additional keys
-s - milliseconds to sleep for DROP FIELD
-S - milliseconds to sleep for CONSTANT DELAY
-P hex_literals_separated - try to recover the key from a conversation sniffed with Proxmark3 (mifarecrack.c based). Accepts several options:
    Concatenated string in hex literal format of form uid:tag_chal:nr_enc:reader_resp:tag_resp
    Example -P 0x5c72325e:0x50829cd6:0xb8671f76:0xe00eefc9:0x4888964f would find key FFFFFFFFFF
-p proxmark3_full.log - tries to parse the log file on it's own (mifarecrack.py based), get the values for option -P and invoke it
-F - tries to fingerprint the input dump (-i) against known cards' data format

drego85@backbox:~$
```

<https://github.com/nfc-tools/mfcuk>



APPLICAZIONI LINUX: HEXCURSE

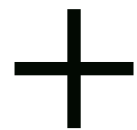
```
Terminale - drego85@backbox: ~/Scrivania
File  Modifica  Visualizza  Terminale  Schede  Aiuto

00000000
00000000  6 37 30 65 24 08 04 00 62 63 64 65 66 67 68 69 C5 40 0B 00 00 00 00
00000017 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000002E 00 00 FF FF FF FF FF FF 07 80 69 FF FF FF FF C5 40 0B 00 00
00000045 00 00 00 00 00 00 00 00 01 00 01 00 00 01 30 00 12 0D 29 23 2A 23 00
0000005C 80 6D 8D 57 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48 F5 5C
00000073 44 66 9D 78 77 88 69 B5 31 28 06 AC 98 00 00 00 00 00 00 00 00 00
0000008A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000A1 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 48 F5 5C 44 66 9D 78 77
000000B8 88 69 B5 31 28 06 AC 98 33 30 31 33 31 30 31 30 36 36 00 00 00 00
000000CF 00 3C A8 83 5B D2 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000E6 00 00 00 00 00 00 00 00 00 00 00 00 48 F5 5C 44 66 9D 78 77 88 69 B5 31 28
000000FD 06 AC 98 4C 49 4E 55 58 20 44 41 59 20 32 30 31 38 2E 00 00 00 00
00000114 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF 00 00
0000012B 00 00 00 00 00 00 48 F5 5C 44 66 9D 78 77 88 69 B5 31 28 06 AC 98 00
00000142 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000159 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000170 48 F5 5C 44 66 9D 78 77 88 69 B5 31 28 06 AC 98 00 00 00 00 00 00
00000187 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000019E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF
000001B5 FF FF 07 80 69 FF FF FF FF FF FF 00 00 00 00 00 00 00 00 00
000001CC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001E3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF FF FF 07 80 69
000001FA FF FF FF FF FF FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000211 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000228 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF FF FF 07 80 69 FF FF FF
0000023F FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000256 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000026D 00 00 00 FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF 00 00 00
00000284 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000029B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF
000002B2 FF FF FF FF FF 07 80 69 FF FF FF FF FF FF 00 00 00 00 00 00 00
000002C9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000002E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF FF FF
000002F7 07 80 69 FF FF FF FF FF FF 00 00 00 00 00 00 00 00 00 00 00
0000030E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000325 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF FF FF 07 80 69 FF FF
0000033C FF FF FF FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000353 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Help  Save  Open  Goto  Find  Hex Addr  Hex Edit  Quit
```

<https://github.com/LonnyGomes/hexcurses>





APPLICAZIONI LINUX: NFC-MFCLASSIC

```
Terminale - drego85@backbox: ~/Scrivania
File Modifica Visualizza Terminale Schede Aiuto
drego85@backbox:~/Scrivania$ nfc-mfclassic
Usage: nfc-mfclassic r|R|w|W a|b <dump.mfd> [<keys.mfd>]
r|R|w|W      - Perform read from (r) or unlocked read from (R) or write to (w) or unlocked write to (W) ca
rd
                *** note that unlocked write will attempt to overwrite block 0 including UID
                *** unlocked read does not require authentication and will reveal A and B keys
                *** unlocking only works with special Mifare 1K cards (Chinese clones)
a|b          - Use A or B keys for action
<dump.mfd>   - MiFare Dump (MFD) used to write (card to MFD) or (MFD to card)
<keys.mfd>   - MiFare Dump (MFD) that contain the keys (optional)
Or: nfc-mfclassic x <dump.mfd> <payload.bin>
x            - Extract payload (data blocks) from MFD
<dump.mfd>   - MiFare Dump (MFD) that contains wanted payload
<payload.bin> - Binary file where payload will be extracted
drego85@backbox:~/Scrivania$
```

<https://github.com/nfc-tools/libnfc>



+ RASPBERRY: DIPENDENZE E PACCHETTI

```
$ sudo apt install autoconf build-essential libtool libusb-dev libpcsclite-dev  
$ sudo apt install libnfc-bin libnfc-examples libnfc-dev
```

Mfoc e **Mfcuk** compilazione una volta scaricati i rispettivi sorgenti:

```
$ autoreconf -vis  
$ ./configure  
$ make  
$ sudo make install
```



+ OPEN SOURCE RFID TOOL COLLECTION

RFID Tools as .deb

- binary .deb Package for Ubuntu x64 only
- containing mfok, fcuk and RFIDLab
- download rfid-tools_1.0.0_amd64.deb
- install prerequisites: `sudo apt-get install pcscd`
- edit `/etc/libccid_Info.plist`
 - `<key>ifdDriverOptions</key>`
 - `<string>0x0004</string>`
- install:
`sudo dpkg -i rfid-tools_1.0.0_amd64.deb`

Bootable USB-Stick image

- bootable Debian USB-Stick image file
- containing mfoc, fcuk, RFIDLab and the cyberflex-shell
- download attachment: rfid_tool_usb_stick.img
- write to USB-Stick:
`sudo dd if=rfid-tools_usb.img of=/dev/sdX bs=4096 count=262144`
- user: root pw: toor

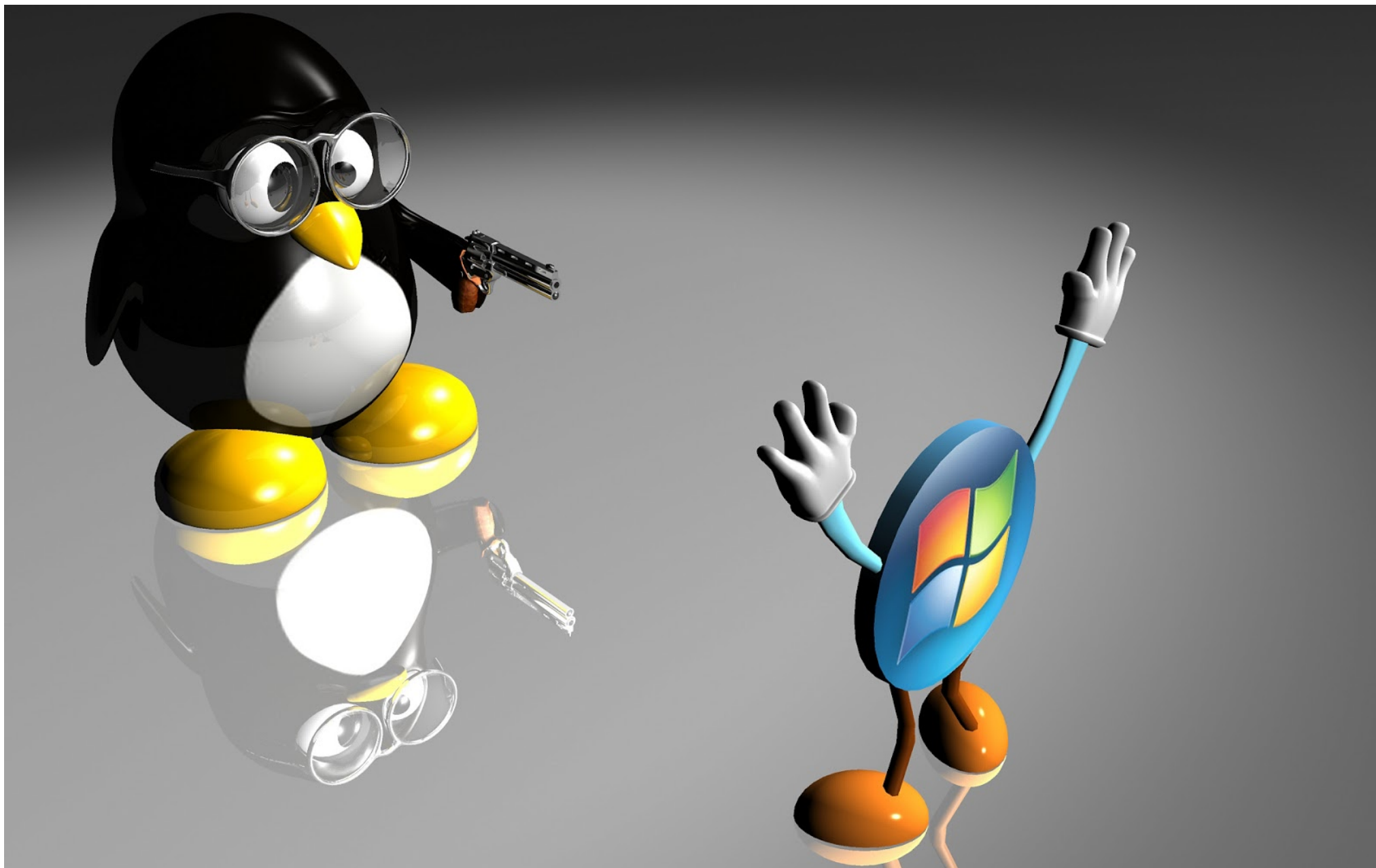
<https://opensource.srlabs.de/projects/rfid>



+ APPLICAZIONI WINDOWS



+ APPLICAZIONI WINDOWS



+ **NFC: LAB MIFARE CLASSIC**

Prendiamo in esempio un tag NFC Mifare Classic 1k

```
$ nfc-list -v
```

```
nfc-list uses libnfc 1.7.1
```

```
NFC device: ACS / ACR122U PICC Interface opened
```

```
[....]
```

```
[....]
```

```
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



+ NFC: LAB MIFARE CLASSIC

Effettuiamo il dump di questo TAG

```
$ mfoc -O original.mfd
```

```
Found Mifare Classic 1k tag
```

```
[....]
```

```
[....]
```

```
Try to authenticate to all sectors with default keys...
```

```
Symbols: '.' no key found, '/' A key found, '\' B key found, 'x' both keys found
```

```
[Key: ffffffff] -> [x.....xxxxxxxxxxx]
```

```
[Key: a0a1a2a3a4a5] -> [x.....xxxxxxxxxxx]
```

```
[Key: d3f7d3f7d3f7] -> [x.....
```



+ NFC: LAB MIFARE CLASSIC

Se il tag non sfrutta le chiavi prestabilite

```
$ mfcuk -C -v 1 -w 6 -R 0:A -O original_ext.dmp -o original.dmp  
mfcuk - 0.3.8
```

Mifare Classic DarkSide Key Recovery Tool - 0.3

by Andrei Costin, zveriu@gmail.com, <http://andreicostin.com>

[....]

[....]

INFO: Connected to NFC reader: ACS / ACR122U PICC Interface

INITIAL ACTIONS MATRIX - UID 56 f7 09 65 - TYPE 0x08 (MC1K)

VERIFY:

Key A sectors: 0 1 2 3 4 5 6 7 8 9 a b c d e f

Key B sectors: 0 1 2 3 4 5 6 7 8 9 a b c d e f



NFC: LAB MIFARE CLASSIC

Visualizziamo e/o Editiamo il dump

```
$ hexcurse original.mfd
```



NFC: LAB MIFARE CLASSIC

Scriviamo il dump editato su una nuova tag

```
$ nfc-mfclassic w B original_edited.mfd newtag.mfd
NFC reader: pn532_uart:/dev/ttyUSB0 opened
Found MIFARE Classic card:
ISO/IEC 14443A (106 kbps) target:
  ATQA (SENS_RES): 00 04
  UID (NFCID1): 8e db 1a 2a
  SAK (SEL_RES): 08
Guessing size: seems to be a 1024-byte card
Writing 64 blocks |..failed to write trailer block 3
X.....|
Done, 62 of 64 blocks written.
```



+ NFC: LAB MIFARE CLASSIC



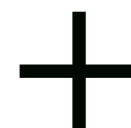
+ Q&A



@AndreaDraghetti



I materiali e i contenuti delle
slide sono protetti da licenza CC BY-NC 3.0



BIBLIOGRAFIA

- Introduzione a NFC {Stefano Sanna} - <http://bit.ly/2ywvl5l>
- Hacking Mifare Classic Cards {Marcio Almedia} - <https://ubm.io/2q7QUEV>
- How to Crack Mifare Classic Cards {FireFart} - <http://bit.ly/2ReiEmW>
-

