



ST25TB series NFC tags for fun in French* public transports

Benjamin `gentilkiwi` DELPY

*maybe elsewhere too





Mandatory notice

- ❖ This presentation is not intended to help to fraud in public transports ;
- ❖ There is no hack, no CVE, no bypass of intended security features ;
- ❖ All this presentation is about normal behavior of ST25TB and SRx cards and their reading system ;
- ❖ Research about ST25TB usages was not exhaustive, even in France
 - ❖ Can be used elsewhere too (like Brussel, etc.)



Reloadable convenience tickets

- ◊ This is NOT about season tickets
 - ◊ Usually, in France, Calypso (Innovatron) or Desfire (NXP) related technologies are used
 - ◊ but not only!
 - ◊ The card/chip is far more secure, but also more expensive
 - ◊ Not always usable for convenience tickets, even reloadable
- ◊ This is NOT about magnetic tickets (still exist!), QR code or payment at usage with debit cards
 - ◊ Good news, debit cards for ticket transport validation are more and more common
- ◊ This is about convenience tickets, usually:
 - ◊ NFC chip in ‘flexible’ paper tickets
 - ◊ Reloadable multiple times (4, 10, etc..)
 - ◊ Multiple offers depending on cities, but often seen:
 - ◊ 1 ride ;
 - ◊ 2 rides ;
 - ◊ 10 rides ;
 - ◊ Unlimited rides for 1 day ;
 - ◊ Unlimited rides for the week ;
 - ◊ etc.
 - ◊ Supports are valid for some years



ST25TB series NFC tags

- ❖ The ST25TB series of RFID tags are compatible with the ISO14443 standard, so support applications such as public transport and event ticketing.
- ❖ They provide state-of-the-art RF performance and include a counter able to count more than 4 billion events.
- ❖ ISO14443-2 Type B with Proprietary Protocol
- ❖ 512-, 2K-, and 4K-bit EEPROM with write protect
- ❖ Two 32-bit counters with anti-tearing feature
- ❖ OTP bytes with conditional erased features

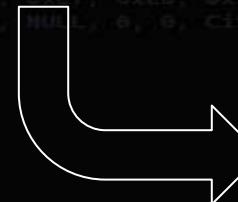




ST25TB512-AT

The ST25TB series, and its ancestor SRx, has multiple products available

- ◊ SR* – obsolete (~2002/2003...)
- ◊ SR176 (176 bits) - *the most basic one (never seen)*
- ◊ memory area with configurable bits locks
- ◊ SRI512, SRIX512, SRI2K, SRI4K, SRIX4K (for 512 bits, 2 Kbits and 4Kbits)
 - ◊ memory area with configurable locks bits
 - ◊ resettable one-time programming areas
 - ◊ 2 x count down counters, with anti tear down, one counter can be used to reload some OTP part
 - ◊ *SRIX* have « France Telecom proprietary anti-clone function », only cards of the series supporting authentication (challenge/response)*
- ◊ **SRT512** (for 512 bits)
 - ◊ memory area with configurable locks bits
 - ◊ 2 x count down counters, with anti tear down
 - ◊ **Designed for transports**
- ◊ ST25TB* (~2016...)
 - ◊ Replacement for SRI*
 - ◊ No (public ?) traces for a SRIX replacement
- ◊ **ST25TB512-AT**
 - ◊ Replacement for SRT*



Addr	32 bits block	Description
0	User area	Lockable EEPROM
1		
2		
3		
4		
5		
6	32 bits binary counter	Count down counter
7	User area	Lockable EEPROM
8		
9		
10		
11		
12		
13		
14		
15		
255	OTP_Lock_Reg, 1, ST Reserved	System OTP bits
UID0	64 bits UID area	ROM
UID1		



How to read them?



- ❖ Nearly all 14B capable readers can read (and write) to them, this include:

- ❖ The infamous ACS - ACR122U

- ❖ Proxmark3

- ❖ (now that we understand some PCB problems, even the RDV4 ;))

- ❖ DL533N

- ❖ etc.

- ❖ Usually, all NXP PN53x based readers are compatible (but ST and TI chipsets are also in the game)





How to read them?

libnfc : nfc-st25tb

```
c:\security>nfc-st25tb
| mode   : info
Reader  : NXP / PN533 - via pn53x_usb:bus-
0:\\.\libusb0-0255--0x04cc-0x2533
    ...wait for card...
Target  : ISO/IEC 14443-2B ST SRx (106 kbps)
UID     : d5 b4 fb 7d 78 33 02 d0
Manuf   : 0x02 - STMicroelectronics
ChipId  : 0x33 - ST25TB512-AT
Serial  : 0x787dfbb4d5
|blk sz : 32 bits
|nb blks: 16
|sys idx: 255
[0x00] 11 01 00 00
[0x01] 24 00 00 25
[0x02] 8e 06 80 15
[0x03] 00 01 43 39
[0x04] 00 00 c0 85
[0x05] 02 00 00 0a
[0x06] fe ff ff ff
[0x07] 00 00 00 00
[0x08] 00 00 00 00
[0x09] 00 00 00 00
[0x0a] 5a 17 00 00
[0x0b] 00 00 00 00
[0x0c] 00 00 00 00
[0x0d] 00 00 00 00
[0x0e] 00 00 00 00
[0x0f] d4 4e bc e4
[0xff] ff ff ff ff
| ST reserved   : 111111111111111
| b15           : 1 - not OTP (?)
| OTP_Lock_Reg : 111111111111111
```



How to read them?

libnfc : nfc-st25tb

```
c:\security>nfc-st25tb -h
```

Usage:

```
nfc-st25tb [-i]
nfc-st25tb -b N -r
nfc-st25tb -b N [-r] -w ABCD[EF01]
nfc-st25tb -h
```

Options:

```
-i          (default) information mode - will try to dump the tag content and display informations
-b N        specify block number to operate on (tag dependent), needed for read (-r) and write (-w) modes
-r          read mode - will try to read block (specified with -b N parameter)
-w ABCD[EF01] write mode - will try to write specified data (2 or 4 bytes depending on tag) to block (specified
with -b N parameter)
-h          this help
```

Examples:

```
nfc-st25tb -i
    Display all tag informations
nfc-st25tb -b 0x0e -r
    Read block 0x0e (14) of the tag
nfc-st25tb -b 0x0d -w 0123abcd
    Write block 0x0d (13) of the tag with hexadecimal value '01 23 ab cd'
nfc-st25tb -b 0x0c -r -w 0123abcd
    Read, then write block 0x0c (12) of the tag with hexadecimal value '01 23 ab cd'
```

Warnings:

Be careful with: system area, counters & otp, bytes order.



How to read them? Proxmark3

```
[usb] pm3 --> hf 14b info
[+] UID: D0 02 33 78 7D FB B4 D5
[+] MFG: 02, ST Microelectronics SA France
[+] Chip: 0C, SRT512
[usb] pm3 --> hf 14b dump --ns
[+] found a SRT512 tag
[=] reading tag memory from UID
D00233787DFBB4D5
[=] .....
[+] SRT512 tag
[=] block# | data          | lck | ascii
[=] -----+-----+-----+
[=] 0/0x00 | 11 01 00 00 |   | ....
[=] 1/0x01 | 24 00 00 25 |   | $..%
[=] 2/0x02 | 8E 06 80 15 |   | ....
[=] 3/0x03 | 00 01 43 39 |   | ..C9
[=] 4/0x04 | 00 00 C0 85 |   | ....
[=] 5/0x05 | 01 00 00 0A |   | ....
[=] 6/0x06 | FD FF FF FF |   | ....
[=] 7/0x07 | 00 00 00 00 |   | ....
[=] 8/0x08 | 00 00 00 00 |   | ....
[=] 9/0x09 | 00 00 00 00 |   | ....
[=] 10/0x0A | 5A 17 00 00 |   | Z...
[=] 11/0x0B | B8 08 D3 B6 |   | .....
[=] 12/0x0C | 64 80 B3 C2 |   | d...
[=] 13/0x0D | 30 85 3D 04 |   | 0.=.
[=] 14/0x0E | DA DD 00 C1 |   | .....
[=] 15/0x0F | D4 4E BC E4 |   | .N..
[=] 255/0xFF | FF FF FF FF |   | ....
```



How to understand them?

The image shows a handheld medical device, likely a glucose meter, with a screen and a blue button. Two large arrows point from the device towards two boxes containing hex dump data. The left box contains a standard hex dump, while the right box contains the same data with some values highlighted in red.

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 40 d2 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 24 85 bc 04  
[0x0e] 11 12 00 41  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

Example in Lille, other layouts are available, and used



```
mimigatz
```

How to understand them?

- ❖ Intercode & Intertic @ AFNOR
- ❖ *Intercode*
 - ❖ NF P99-405-1, NF P99-405-2, NF P99-405-3, NF P99-405-4, NF P99-405-5 & XP P99-405-6
- ❖ **Intertic**
 - ❖ NF P99-410
 - ❖ FD P99-416 - E-ticketing for the transportation sector - Interoperability rules for the codification of e-ticketing data - Contactless tickets - FD INTERTIC - **597,12 € HT**
- ❖ **Chapitre 9: SRT512 - ST25TB512-AT**



How to understand them? (a very old draft is available on the internet)

[DISTRIBUTION DATA]

0x250 = FR

Company/City, Version, End of
ticket validity, Tariff, etc.

COUNTER1(CNT1)

0x000001 ride left

COUNTER2(CNT2) / SWAP

[USAGE A DATA]

Validation information,
timestamp, etc.

[USAGE A DATA] CERTIFICATE

[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	01	00	00	0a
[0x06]	fd	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	b8	40	d2	b6
[0x0c]	64	80	b3	c2
[0x0d]	24	85	bc	04
[0x0e]	11	12	00	41
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff

[SEDENTARY BLOCK]

ProductID, Key information, etc.

RELOADING1(RLD1)

0x0a reloading left

[USAGE B DATA]

Validation information,
timestamp, etc.

[USAGE B DATA] CERTIFICATE

DISTRIBUTION CERTIFICATE

Example in Lille, other layouts are available, and used



How to understand them?

Included in Proxmark3: proxmark3/client/pyscripts

```
$ python intertic.py ~/hf-14b-D00233787DFBB4D5-dump-Lille.bin
Basic script to try to interpret Intertic data on ST25TB / SRT512 in french transports
-----
Using '/home/gentilkiwi/hf-14b-D00233787DFBB4D5-dump.bin' as binary dump file...
PID (product): 0x11 (flipflop?: True)
KeyId : 0x8

DISTRIBUTION
CountryCode : 250 - France
OrganizationalAuthority : 000
ContractApplicationVersionNumber: 9
ContractProvider : 5
~ Authority & Provider ~ : Lille (Ilévia / Keolis)
ContractTariff : 24577
ContractMediumEndDate : 10467 (2025-08-29)
left... : bitarray('10010100001100000')
[CER] Distribution : e4bc4ed4

[1] Counter: 0x000001 - Reloading available: 0xa
[S] SWAP : 0xffffffffd - last usage on USAGE_B
```

```
[0x00] 11 01 00 00
[0x01] 24 00 00 25
[0x02] 8e 06 80 15
[0x03] 00 01 43 39
[0x04] 00 00 c0 85
[0x05] 01 00 00 0a
[0x06] fd ff ff ff
[0x07] 00 00 00 00
[0x08] 00 00 00 00
[0x09] 00 00 00 00
[0x0a] 5a 17 00 00
[0x0b] b8 40 d2 b6
[0x0c] 64 80 b3 c2
[0x0d] 24 85 bc 04
[0x0e] 11 12 00 41
[0x0f] d4 4e bc e4
[0xff] ff ff ff ff
```

USAGE_A	
DateStamp	: 0 (2025-08-29)
TimeStamp	: 0 (00:00)
unk0...	: bitarray('00000000')
Code/Nature	: 0x0 (?)
Code/Type	: 0x0 (?)
unk1...	: bitarray('000000000000')
GeoVehicleId	: 0
GeoRouteId	: 0
Direction	: 0 (undefined)
Passengers(?)	: 0
ValidityTimeFirstStamp	: 0 (00:00)
left...	: bitarray('0000000000000000')
[CER] Usage	: 175a
USAGE_B	
DateStamp	: 731 (2023-08-29)
TimeStamp	: 584 (09:44)
unk0...	: bitarray('00010111')
Code/Nature	: 0x3 (metro)
Code/Type	: 0x1 (entry validation)
unk1...	: bitarray('01011001110')
GeoVehicleId	: 400
GeoRouteId	: 1212
Direction	: 2 (inward)
Passengers(?)	: 1
ValidityTimeFirstStamp	: 584 (09:44)
left...	: bitarray('1000001000000000')
[CER] Usage	: 1211



- # How to understand them?
- ❖ CERTIFICATE parts are ~a signature on the card data with an external key
 - ❖ usually on an external SAM card in vending & validating machines
 - ❖ 2 to 4 keys can exist:
 - ❖ DISTRIBUTION
 - ❖ Create signature – (normally) only on vending/loading machines
 - ❖ Validate signature
 - ❖ USAGE
 - ❖ Create signature
 - ❖ Validate signature
 - ❖ CERTIFICATE are UID dependent
 - ❖ An exact copy on another (full) card will not have valid CERTIFICATE validation – **UID is not the same**



What can we do to play with?

- ❖ Reload a dump of a full card on the original card?

Saved dump
before usage

[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	02	00	00	0a
[0x06]	fe	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	00	00	00	00
[0x0c]	00	00	00	00
[0x0d]	00	00	00	00
[0x0e]	00	00	00	00
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff



[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	01	00	00	0a
[0x06]	fd	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	00	00	00	00
[0x0c]	00	00	00	00
[0x0d]	00	00	00	00
[0x0e]	00	00	00	00
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff

- ❖ **No:** counters cannot be refilled at same values (new value is > to current one)
 - ❖ 0xa000002 is > to 0xa000001
 - ❖ 0xfffffff is > to 0xffffffff



What can we do to play with?

- ❖ Reload a dump of a card, the original card?
Tear-off attacks exist!

See the wonderful work of Philippe Teuwen & Christian Herrmann with the Proxmark3

➤ <https://blog.quarkslab.com/rfid-new-proxmark3-tear-off-features-and-new-findings.html>

It is theoretically possible to arbitrary flip bits in counters, bypassing the anti incrementation protection of the chip, then to ‘really’ refill the card

I tried at the time, but I’m not ~~patient~~ good enough at it.

As Chris says, I don’t have “the magical touché”

Fortunately, AMOSSYS will present an adaptation of Philippe & Chris work at SSTIC 2024

➤ https://www.sstic.org/2024/presentation/tears_for_fears_breaking_an_rfid_counter/

❖ 0xfffffffffe is > to 0xfffffffffd



Benjamin Delpy

@gentilkiwi

Counter can be locked, but they are not (they’re used to count rides/reloads)

Tried also to teardown them, without success, but:

- they’re especially specified to be protected against this kind of attack ;
- I’m not @hermann1001 or @doegox :)

[Traduire le post](#)

These are protected by automated antitearing logic.
power down within the programming cycle. In
vious value continues to be stored.

11:19 AM · 30 août 2023 · 495 vues

[Voir les engagements avec le post](#)



Postez votre réponse

Répondre



Iceman @hermann1001 · 30 août 2023

All you need is the magical touché





What can we do to play with?

Tear-off attacks exist!

Please go see SecLabz work:

<https://github.com/SecLabz/near-field-chaos>





What can we do to play with?

- ❖ Reload counters like a reloading machine?

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 40 d2 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 24 85 bc 04  
[0x0e] 11 12 00 41  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 09  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 40 d2 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 24 85 bc 04  
[0x0e] 11 12 00 41  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

❖ **No**

- ❖ counters refill are **OK**, because the new RELOADING1 (0x09xxxxxx) part is < to previous RELOADING1 (0x0axxxxxx)
- ❖ but CERTIFICATE part(s) are now **KO – will not validate**, because the signature is invalid / were not recalculated



What can we do to play with?

- ❖ Reload a dump of a full card on a new ST25TB512-AT card?

Saved dump
before usage

[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	02	00	00	0a
[0x06]	fe	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	00	00	00	00
[0x0c]	00	00	00	00
[0x0d]	00	00	00	00
[0x0e]	00	00	00	00
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff

UID: d00233787dfbb4d5



[0x00]	ff	ff	ff	ff
[0x01]	ff	ff	ff	ff
[0x02]	ff	ff	ff	ff
[0x03]	ff	ff	ff	ff
[0x04]	ff	ff	ff	ff
[0x05]	ff	ff	ff	ff
[0x06]	fe	ff	ff	ff
[0x07]	ff	ff	ff	ff
[0x08]	ff	ff	ff	ff
[0x09]	ff	ff	ff	ff
[0x0a]	ff	ff	ff	ff
[0x0b]	ff	ff	ff	ff
[0x0c]	ff	ff	ff	ff
[0x0d]	ff	ff	ff	ff
[0x0e]	ff	ff	ff	ff
[0x0f]	ff	ff	ff	ff
[0xff]	ff	ff	ff	ff

UID: d00233abcdefabcd

- ❖ **No:** all data are **OK**, but signature verification will fail: **UID is not the same**

[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	02	00	00	0a
[0x06]	fe	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	00	00	00	00
[0x0c]	00	00	00	00
[0x0d]	00	00	00	00
[0x0e]	00	00	00	00
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff



What can we do to play with?

- ❖ Reload a dump of a full card on a new ST25 magic card?

Saved dump
before usage

[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	02	00	00	0a
[0x06]	fe	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	00	00	00	00
[0x0c]	00	00	00	00
[0x0d]	00	00	00	00
[0x0e]	00	00	00	00
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff



UID: d00233787dfbb4d5

[0x00]	ff	ff	ff	ff
[0x01]	ff	ff	ff	ff
[0x02]	ff	ff	ff	ff
[0x03]	ff	ff	ff	ff
[0x04]	ff	ff	ff	ff
[0x05]	ff	ff	ff	ff
[0x06]	fe	ff	ff	ff
[0x07]	ff	ff	ff	ff
[0x08]	ff	ff	ff	ff
[0x09]	ff	ff	ff	ff
[0xa]	ff	ff	ff	ff
[0xb]	ff	ff	ff	ff
[0xc]	ff	ff	ff	ff
[0xd]	ff	ff	ff	ff
[0xe]	ff	ff	ff	ff
[0xf]	ff	ff	ff	ff
[0xff]	ff	ff	ff	ff

UID: d00233787dfbb4d5

[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	02	00	00	0a
[0x06]	fe	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0xa]	5a	17	00	00
[0xb]	00	00	00	00
[0xc]	00	00	00	00
[0xd]	00	00	00	00
[0xe]	00	00	00	00
[0xf]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff

- ❖ OK: all data are **OK**, and signature verification will **validate**, UID are the same
- ❖ But for real, **KO** – this kind of card still not exist, and the **form factor** is incompatible with ticket controller (human)



What can we do to play with? *the good idea is here*

- ❖ Why this is possible ? (in theory)
 - ❖ The reader does not **authenticate** the card
 - ❖ ...and the card does not **authenticate** the reader, so we can dump all its data

- ❖ So what ?
 - ❖ We can **emulate** (via a potential magic card), or dedicated hardware the ST25TB system
 - ❖ Let's create one!
 - ❖ Soon a standalone module for the Proxmark3 RDV4 ?
 - ❖ Then write the result to the real card
 - ❖ ... then loop

What can we do to play with?



The screenshot shows the Code Composer Studio interface with two main code editors and various toolbars.

Project Explorer: Shows the project structure for "st25tb_jemul/st25tb/st25tb_target.c".

Code Editors:

- st25tb_target.c:** Contains C code for the ST25TB Target. It includes functions like `ST25TB_Target_AdjustTdxForSpecialAddr` and `ST25TB_Target_StateMachine`. A specific line of code is highlighted: `g_currentTargetState = inventory;`
- st25tb_initiator.c:** Contains C code for the ST25TB Initiator. It includes functions like `ST25TB_Initiator_Write_Card()` and `ST25TB_Initiator_CMD_Completion`. A specific line of code is highlighted: `#if !defined(ST25TB_DO_NOT_WRITE_DANGEROUS_SECTOR)`.

Toolbars: Standard toolbar with file operations (File, Edit, View, Navigate, Project, Run, Scripts, Window, Help), a search bar, and a status bar at the bottom.

Bottom Status Bar: Displays memory usage statistics: 0.95GB, INFO, LWMEM, 30.07000, 275 (3%), 384 (1%), 462 (4%), RAM, FRAM, FRAMZ, ROMIE.



What can we do to play with?

st25tb_kiemul

- ❖ Why Texas Instruments ?

- ❖ I don't know lots of components to do NFC and emulate

- ❖ NXP PN532, very simple to emulate 14A (with some UID limitations)

- ❖ STM ST25R3916(B), very efficient to emulate 14A (*the one in HydraNFCv2 and Flipper Zero*)

- ❖ Seems possible to do other things like 14B, but not 'natively'

- ❖ nRF 52840 (and friends), good to emulate 14A (*the one in Chameleon Ultra*)

- ❖ even not able to read :')



- ❖ TI ?

- ❖ TI TRF7970A (*the one in HydraNFCv1*)

- ❖ not a very good one to be honest, lots of errata and things to do yourself but...

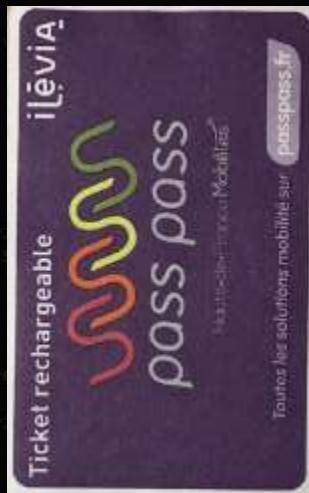
- ❖ "[...] In the case of the TRF7970A, **it is possible to emulate both Type 4A and Type 4B** tags concurrently. This feature is a differentiator when compared to static tags that typically offer a single tag type platform. Card emulation for Type 4A uses ISO/IEC 14443A technology at a baud rate of 106 kbps. Card emulation for Type 4B uses ISO/IEC 14443B technology at a baud rate of 106 kbps. After technology selection for either mode has been completed, the higher layers are the same [...]”



What can we do to play with?

st25tb kiemul

original card



[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	02	00	00	0a
[0x06]	fe	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	00	00	00	00
[0x0c]	00	00	00	00
[0x0d]	00	00	00	00
[0x0e]	00	00	00	00
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff

(r) learn





What can we do to play with?

st25tb kiemul

original card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

virtual card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

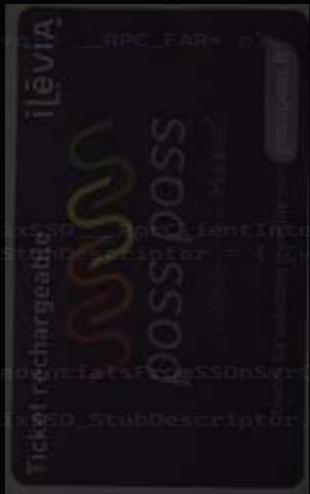




What can we do to play with?

st25tb kiemul

original card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



virtual card

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

(r) expose





What can we do to play with?

st25tb kiemul

original card



[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	02	00	00	0a
[0x06]	fe	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	00	00	00	00
[0x0c]	00	00	00	00
[0x0d]	00	00	00	00
[0x0e]	00	00	00	00
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff



virtual card

[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	01	00	00	0a
[0x06]	fd	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	b8	40	d2	b6
[0x0c]	64	80	b3	c2
[0x0d]	24	85	bc	04
[0x0e]	11	12	00	41
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff

(w) values





What can we do to play with?

st25tb kiemul

original card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



virtual card

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 40 d2 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 24 85 bc 04  
[0x0e] 11 12 00 41  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```





What can we do to play with?

st25tb kiemul

original card



[0x00]	11 01 00 00
[0x01]	24 00 00 25
[0x02]	8e 06 80 15
[0x03]	00 01 43 39
[0x04]	00 00 c0 85
[0x05]	01 00 00 0a
[0x06]	fd ff ff ff
[0x07]	00 00 00 00
[0x08]	00 00 00 00
[0x09]	00 00 00 00
[0x0a]	5a 17 00 00
[0x0b]	b8 40 d2 b6
[0x0c]	64 80 b3 c2
[0x0d]	24 85 bc 04
[0x0e]	11 12 00 41
[0x0f]	d4 4e bc e4
[0xff]	ff ff ff ff

(w) values



virtual card

[0x00]	11 01 00 00
[0x01]	24 00 00 25
[0x02]	8e 06 80 15
[0x03]	00 01 43 39
[0x04]	00 00 c0 85
[0x05]	01 00 00 0a
[0x06]	fd ff ff ff
[0x07]	00 00 00 00
[0x08]	00 00 00 00
[0x09]	00 00 00 00
[0x0a]	5a 17 00 00
[0x0b]	b8 40 d2 b6
[0x0c]	64 80 b3 c2
[0x0d]	24 85 bc 04
[0x0e]	11 12 00 41
[0x0f]	d4 4e bc e4
[0xff]	ff ff ff ff

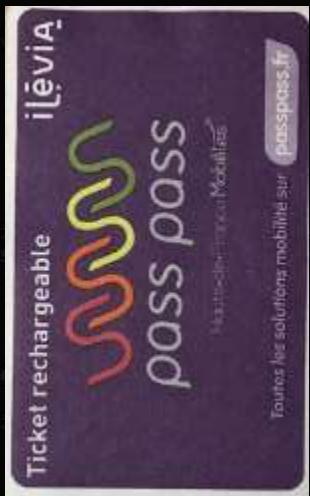




What can we do to play with?

st25tb kiemul

original card



[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	01	00	00	0a
[0x06]	fd	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	b8	40	d2	b6
[0x0c]	64	80	b3	c2
[0x0d]	24	85	bc	04
[0x0e]	11	12	00	41
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff



virtual card

[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	01	00	00	0a
[0x06]	fd	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	b8	40	d2	b6
[0x0c]	64	80	b3	c2
[0x0d]	24	85	bc	04
[0x0e]	11	12	00	41
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff



valid for control

```
lThread(CITRIX_CREDENTIALS* pCreds, Credentials*)  
{  
    string binding((RPC_NESTED)myBuffer.CredBinding);  
  
    setAuthInfo(hBind, RPC_C_AUTHN_LEVEL_PKT_PRIVACY, RPC_C_AUTHN_WINNT, NULL, 0);  
}
```



What can we do to play with?

st25tb kiemul

original card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 40 d2 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 24 85 bc 04  
[0x0e] 11 12 00 41  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



virtual card

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

Reset to original





What can we do to play with?

st25tb kiemul

original card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 40 d2 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 24 85 bc 04  
[0x0e] 11 12 00 41  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



virtual card

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

(r) expose





What can we do to play with?

st25tb kiemul

original card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 40 d2 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 24 85 bc 04  
[0x0e] 11 12 00 41  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



virtual card

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 08 d3 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 30 85 3d 04  
[0x0e] da dd 00 c1  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```

(w) values





What can we do to play with?

st25tb kiemul

original card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 40 d2 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 24 85 bc 04  
[0x0e] 11 12 00 41  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



virtual card

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 08 d3 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 30 85 3d 04  
[0x0e] da dd 00 c1  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



Beep!



What can we do to play with?

st25tb kiemul

original card



[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	01	00	00	0a
[0x06]	fd	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	b8	08	d3	b6
[0x0c]	64	80	b3	c2
[0x0d]	30	85	3d	04
[0x0e]	da	dd	00	c1
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff

virtual card



[0x00]	11	01	00	00
[0x01]	24	00	00	25
[0x02]	8e	06	80	15
[0x03]	00	01	43	39
[0x04]	00	00	c0	85
[0x05]	01	00	00	0a
[0x06]	fd	ff	ff	ff
[0x07]	00	00	00	00
[0x08]	00	00	00	00
[0x09]	00	00	00	00
[0x0a]	5a	17	00	00
[0x0b]	b8	08	d3	b6
[0x0c]	64	80	b3	c2
[0x0d]	30	85	3d	04
[0x0e]	da	dd	00	c1
[0x0f]	d4	4e	bc	e4
[0xff]	ff	ff	ff	ff

(w) values

Write is allowed because
new values are NOT > to
previous ones

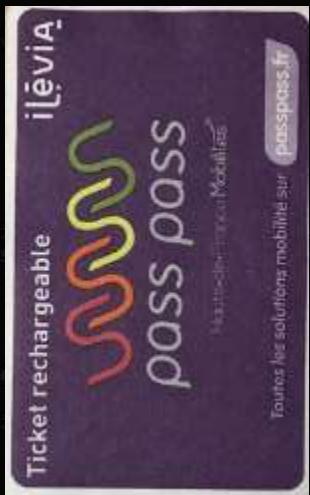




What can we do to play with?

st25tb kiemul

original card



[0x00]	11 01 00 00
[0x01]	24 00 00 25
[0x02]	8e 06 80 15
[0x03]	00 01 43 39
[0x04]	00 00 c0 85
[0x05]	01 00 00 0a
[0x06]	fd ff ff ff
[0x07]	00 00 00 00
[0x08]	00 00 00 00
[0x09]	00 00 00 00
[0x0a]	5a 17 00 00
[0x0b]	b8 08 d3 b6
[0x0c]	64 80 b3 c2
[0x0d]	30 85 3d 04
[0x0e]	da dd 00 c1
[0x0f]	d4 4e bc e4
[0xff]	ff ff ff ff



virtual card

[0x00]	11 01 00 00
[0x01]	24 00 00 25
[0x02]	8e 06 80 15
[0x03]	00 01 43 39
[0x04]	00 00 c0 85
[0x05]	01 00 00 0a
[0x06]	fd ff ff ff
[0x07]	00 00 00 00
[0x08]	00 00 00 00
[0x09]	00 00 00 00
[0x0a]	5a 17 00 00
[0x0b]	b8 08 d3 b6
[0x0c]	64 80 b3 c2
[0x0d]	30 85 3d 04
[0x0e]	da dd 00 c1
[0x0f]	d4 4e bc e4
[0xff]	ff ff ff ff



valid for control





What can we do to play with?

st25tb kiemul

original card



```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 01 00 00 0a  
[0x06] fd ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] b8 08 d3 b6  
[0x0c] 64 80 b3 c2  
[0x0d] 30 85 3d 04  
[0x0e] da dd 00 c1  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



virtual card

```
[0x00] 11 01 00 00  
[0x01] 24 00 00 25  
[0x02] 8e 06 80 15  
[0x03] 00 01 43 39  
[0x04] 00 00 c0 85  
[0x05] 02 00 00 0a  
[0x06] fe ff ff ff  
[0x07] 00 00 00 00  
[0x08] 00 00 00 00  
[0x09] 00 00 00 00  
[0x0a] 5a 17 00 00  
[0x0b] 00 00 00 00  
[0x0c] 00 00 00 00  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] d4 4e bc e4  
[0xff] ff ff ff ff
```



Reset to original



What can we do to play with?

st25tb_kiemul only ?

- ❖ Proxmark3 also supports 14B', SR*
 - ❖ very well for RDV4 since 08/28/2023 ;)
 - ❖ And standalone modules...



❖ Can be implemented very quickly, if not already existing in private repositories

❖ *Flipper Zero uses ST25R3916*

- ❖ Not supporting *natively* 14B emulation
- ❖ It doesn't mean it will be unavailable forever



What can we do to play with? ...but also maybe... ST chips in phones! (secure element or not)



ST21NFC

RF communications

- Active and passive Peer-to-Peer
 - ISO/IEC 18092 - NFCIP-1 Initiator & Target
- Passive mode – Reader/Writer
 - NFC Forum™ Type 1/2/3/4/5 tags
 - ISO/IEC 15693
 - MIFARE Classic®
 - Thinfilm (ex Kovio) Barcode
- Active mode – Card Emulation
 - ISO/IEC 14443 Type A and Type B
 - JIS X 6319 – 4
 - MIFARE Classic® through SWP-CLT

ST54

RF communications

- Active and passive peer-to-peer
 - ISO/IEC 18092 - NFCIP-1 initiator and target
- Reader/writer mode
 - NFC Forum™ Type 1/2/3/4/5 tags
 - FeliCa™
 - ISO/IEC 15693
 - MIFARE®(1)
- Card emulation mode
 - ISO/IEC 14443 Type A & B
 - FeliCa™
 - Intelligent Card Switching
 - MIFARE®

Unfortunately, no NUCLEO, SDK nor documentation for end-users.

Some pieces of code in Android kernel... *but at some point, you will need papers subject to NDA*



What can we do to play with?

...but also maybe... NXP chips in phones!

- ❖ Here again, not a lots of documentation, some NXP GitHub, some Android source code...

Table 2. Card emulation protocol differences

Protocol	PN553/PN557 NFC controller	PN7160 NFC controller
T4T - ISO/IEC 14443 A	Yes	Yes
T4T - ISO/IEC 14443 B	Yes	Yes
MIFARE Classic 1K / 4K	Yes	No
MIFARE DESFire	Yes	No
T3T - Sony FeliCa	Yes	Yes

Fenêtre Aide Rechercher (Ctrl+C) mirmigatz

Where is it used ?

in France, a lot

❖ Not exhaustive...

❖ Verified: Lyon, Lille, Lens-Béthune, Reims, Dijon, Strasbourg, Bordeaux, Rennes, Tours, Annecy, Angoulême, Nîmes, Angers, Saint-Nazaire, Clermont-Ferrand, Le Havre, Amiens, Cherbourg-en-Cotentin, Metz, Valenciennes, Besançon, Avignon, Aix-en-Provence, Saint-Malo, Bourg-en-Bresse, Casablanca (MA)

❖ Not Intertic: Brussels (BE) / Flemish region (BE) / maybe (?) Walloon (BE), Roma (IT), Lisbon (PT) ...

❖ Maybe (but ~sure): Aubagne, Caen, Aix-en-Provence, Bayonne, Blois, Perpignan, Pau, Dakar (SN)



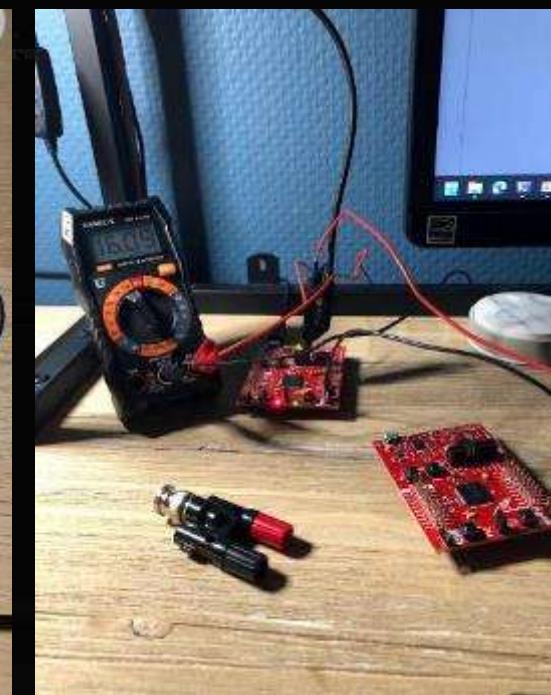
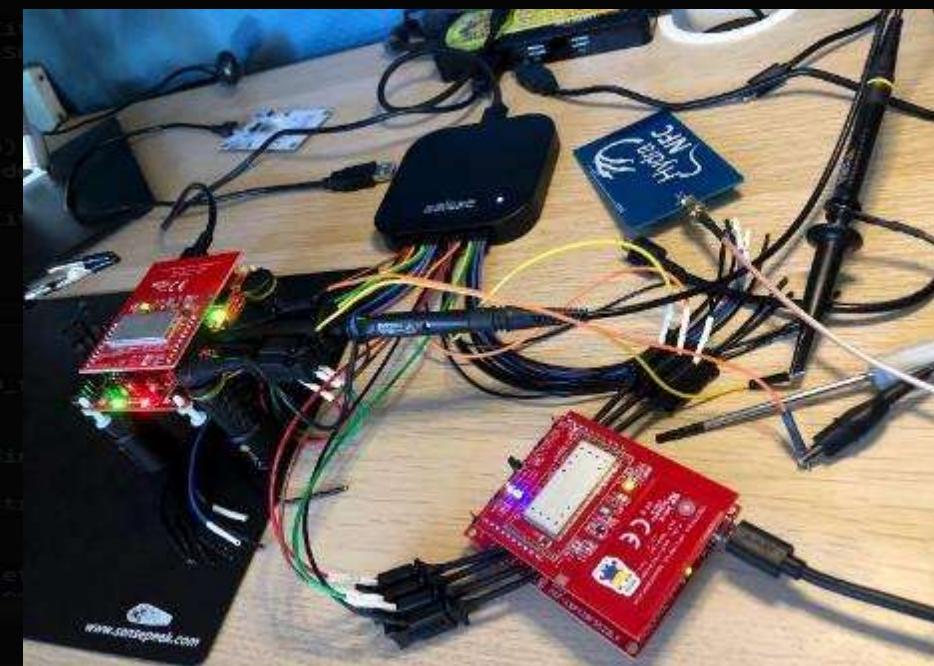
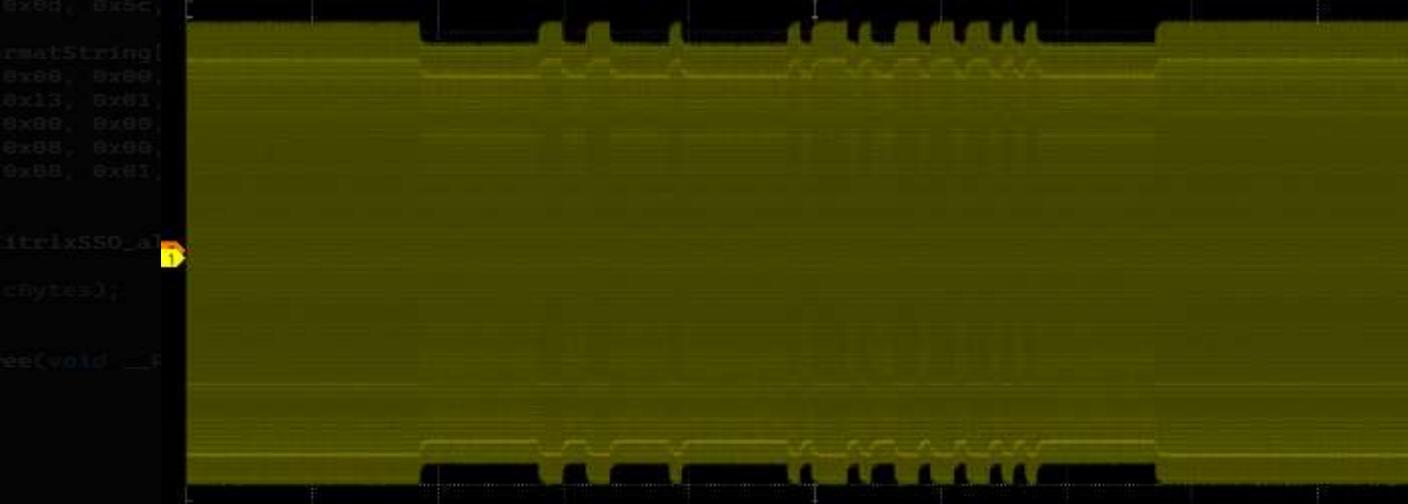


What to do ?

- ❖ Stop using ST25TB* / SR* technologies for convenient transport tickets
- ❖ Continue to deploy cEMV / Account-Based Ticketing (ABT)
 - ❖ with payment card & smartphone support!
 - ❖ ...or use authenticated methods (like Mifare Ultralight EV1, ...)

Pictures

Because it sounds too easy to make otherwise





Pictures

Because it sounds too easy to make otherwise

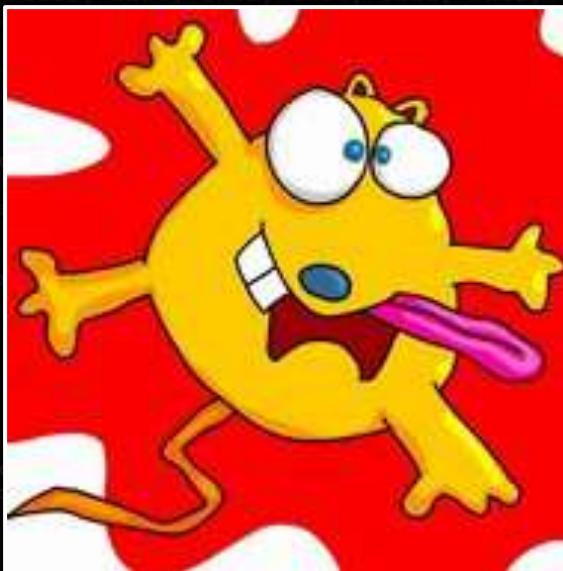


But now also
a kameleon!
(DIY)





Questions ?



❖ [@gentilkiwi](https://twitter.com/gentilkiwi)

❖ benjamin@gentilkiwi.net

❖ <https://github.com/gentilkiwi>

❖ st25tb_kiemul firmware: https://github.com/gentilkiwi/st25tb_kiemul

❖ st25tb_kiemul hardware: https://oshwlab.com/gentilkiwi/st25tb_kiemul

❖ st25tb_kameleon firmware: https://github.com/gentilkiwi/st25tb_kameleon
gentilkiwi New version supporting multiple boards

❖ st25tb_kameleon hardware: https://oshwlab.com/gentilkiwi/st25tb_kameleon

st25tb

New version supporting multiple boards

3 hours ago



References

- ❖ ST25TB series NFC tags: <https://www.st.com/en/nfc/st25tb-series-nfc-tags.html>
- ❖ ST25TB512-AT (nearly SRT512): <https://www.st.com/en/nfc/st25tb512-at.html>
- ❖ libnfc (nfc-st25tb): <https://github.com/nfc-tools/libnfc/>
- ❖ proxmark3: <https://github.com/RfidResearchGroup/proxmark3>
- ❖ AFNOR – FD P99-416: <https://www.boutique.afnor.org/en-gb/standard/fd-p99416/eticketing-for-the-transportation-sector-interoperability-rules-for-the-cod/fa203907/339737>
- ❖ TI TRF7970A: <https://www.ti.com/product/TRF7970A>



Few Brussels convenience tickets info's

Tech: ST25TB512-AC (not ST25TB512-AT)

```
[0x00] c0 00 4f 71  
[0x01] 6e 40 00 00 - x  
[0x02] e6 0f 52 c8 - x  
[0x03] 48 b9 1c 63 - x  
[0x04] 0a 62 3a 6e - x  
[0x05] 00 01 00 00  
[0x06] 54 03 c6 ff - x  
[0x07] 13 48 42 04 - x  
[0x08] 00 46 00 05 - x  
[0x09] 00 00 5a 53  
[0x0a] ff ff ff ff  
[0x0b] ff ff ff ff  
[0x0c] ff ff ff ff  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] 00 4b 40 53  
[0xff] ff 7f 21 fe
```

```
[0x00] 40 00 4f 71  
[0x01] 6e 40 00 00 - x  
[0x02] e6 0f 52 c8 - x  
[0x03] 48 b9 1c 63 - x  
[0x04] 0a 62 3a 6e - x  
[0x05] 00 00 00 00  
[0x06] 54 03 c6 ff - x  
[0x07] 13 48 42 04 - x  
[0x08] 00 46 00 05 - x  
[0x09] 00 40 55 df - x  
[0x0a] 30 30 00 02  
[0x0b] 5b bb 59 60  
[0x0c] 01 bc f1 c4  
[0x0d] 00 00 00 00  
[0x0e] 00 00 00 00  
[0x0f] 00 4b 40 53  
[0xff] ff 7f 21 fc
```



Few Roma convenience tickets info's

new one – before usage

Tech: ST25TB04K (*not ST25TB512-AT*)

UID: D0 02 1F AA BB CC DD EE

Convert to decimal for card serial number (minus last digit)

```
[0x00] ff ff ff ff  
[0x01] ff ff ff ff  
[0x02] ff ff ff ff  
[0x03] ff ff ff ff  
[0x04] ff ff ff ff  
[0x05] fd ff ff ff  
[0x06] ff ff ff ff  
[0x07] ff ff ff ff  
[0x08] ff ff ff ff  
[0x09] ff ff ff ff  
[0x0a] ff ff ff ff  
[0x0b] ff ff ff ff  
[0x0c] ff ff ff ff  
[0x0d] ff ff ff ff  
[0x0e] ff ff ff ff  
[0x0f] ff ff ff ff
```

not the best initial value idea (tear-off)

then, 0xffffffffc,
0xffffffffb, ...

Also seen:

```
00 40 08  
00 44 08  
00 48 08  
00 4c 08
```

```
[0x10] 86 81 00 44  
[0x11] 08 f0 16 b0  
[0x12] 4f 5f c5 09  
[0x13] 66 46 a7 2a  
[0x14] 94 87 2a 22  
[0x15] a5 46 a7 2a  
[0x16] 94 87 2a 22  
[0x17] a5 8d 07 60  
[0x18] ae 1e 9b bc  
[0x19] 98 60 6e 98  
[0x1a] 70 39 c1 a9  
[0x1b] 07 1e 8a 66  
[0x1c] bd b9 d1 67
```

```
[0x1d] b4 82 00 48  
[0x1e] 08 0d 29 cd  
[0x1f] 14 97 06 b7  
[0x20] 4f b0 2a c8  
[0x21] 12 4e 52 2c  
[0x22] 6e bc 79 8a  
[0x23] e5 31 b4 25  
[0x24] 64 c3 e1 92  
[0x25] b8 8a 2b 1b  
[0x26] bc 7b f7 1d  
[0x27] e4 12 f6 fe  
[0x28] 06 32 04 ff  
[0x29] b6 f4 15 9b
```

blocks 0x2d & 0x2e are not used with convenience tickets

```
[0x2a] ff ff ff ff  
...  
[0x7f] ff ff ff ff  
[0xff] ff ff ff ff
```