



ST25TB series NFC tags for fun in French* public transports

Benjamin *`gentilkiwi`* DELPY

**maybe elsewhere too*

Unnecessary notice



◆ Benjamin DELPY

◆ Banque de France – DGSi / DIT / ARCOS

◆ Still not CISSP, CISA, OSCP, CHFI, CEH, ISO*, MCSA, CHFI, PASSI... or MVP ;)

◆ Kiwi during the night (and a little bit the day)

◆ Research & Development Security Center (CRDS)





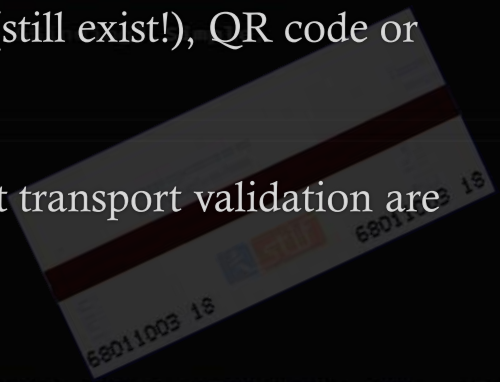
Mandatory notice

- ◆ 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.)

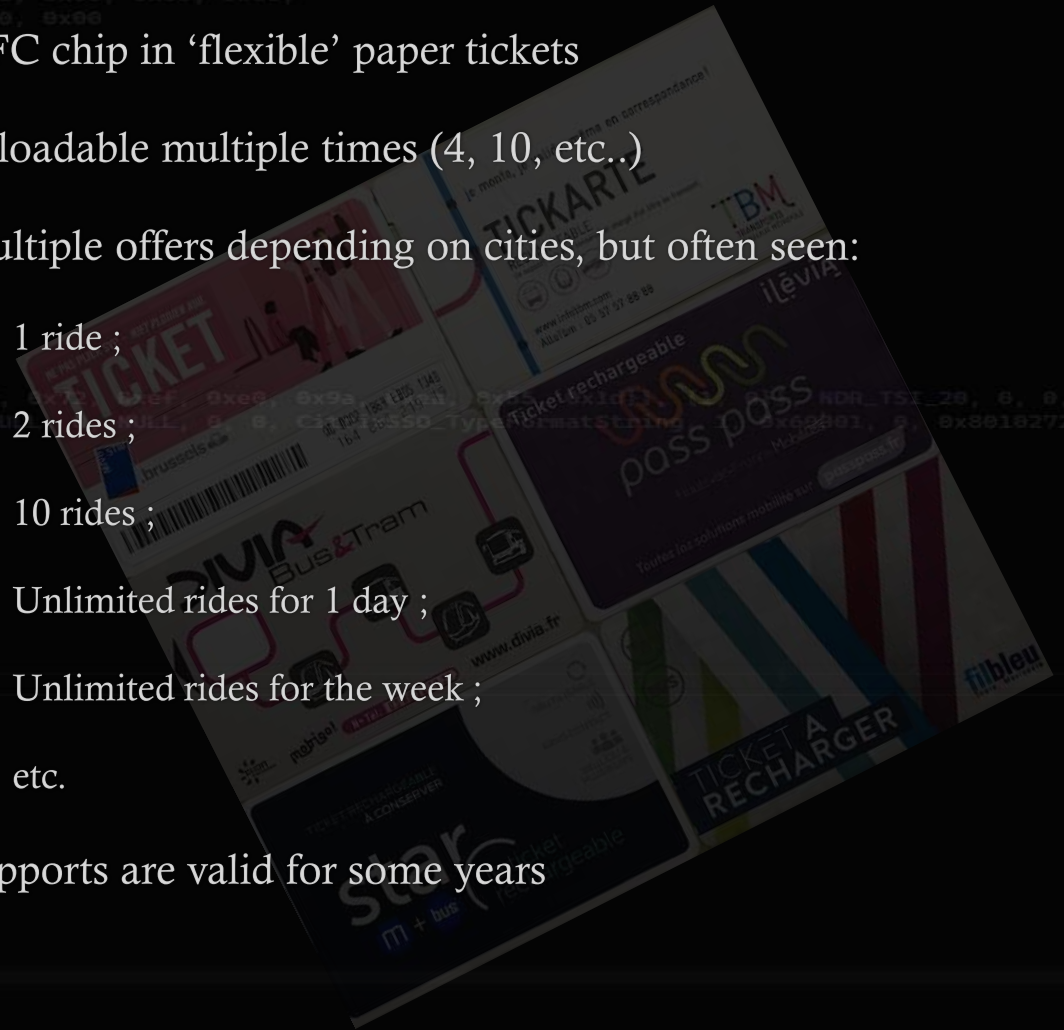


Reloadable convenience tickets

- ◆ 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

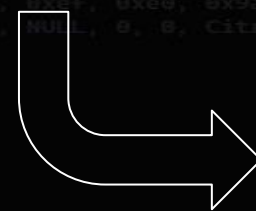




ST25TB512-AT

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...)
 - ◆ ST25TB512-AC, ST25TB02K-AC, ST25TB04K-AC
 - ◆ 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	32 bits binary counter	Count down counter
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

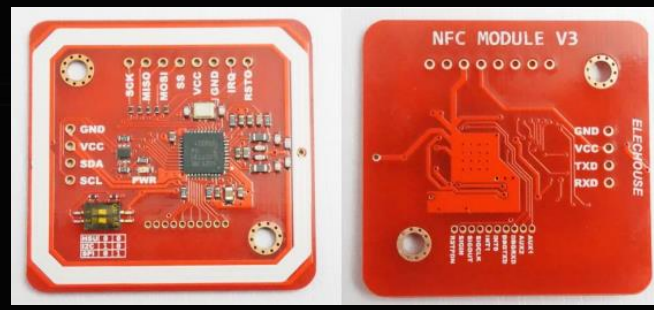
◆ Proxmark3

◆ (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   : 1111111111111111
| b15           : 1 - not OTP (?)
| OTP_Lock_Reg : 1111111111111111
```



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 specied 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 | | ....
[=] -----+-----+---+-----

```

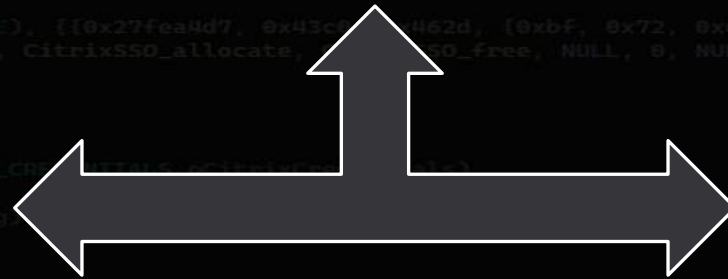



How to understand them?



```
[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



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)
 ?

? CERTIFICATE ?

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

[SEEDENTARY BLOCK]
 ProductID, Key information, etc.

RELOADING1(RLD1)
 0x0a reloading left

RELOADING2(RLD2)
 ?

[USAGE DATA]
 Validation information,
 timestamp, etc.

DISTRIBUTION CERTIFICATE



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)

◆ 0x0a000002 is > to 0x0a000001

◆ 0xfffffffffe is > to 0xfffffffffd



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
```

*New blank / full
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
```

UID: *d00233abcdefabcd*

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

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
[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
```

★ ★ ★
magic 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
```

UID: *d00233787dfbb4d5*

- ◆ **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?

st25tb_kiemul



```
workspace_v12 - st25tb_kiemul/st25tb/st25tb_target.c - Code Composer Studio
File Edit View Navigate Project Run Scripts Window Help
Project Explorer
st25tb_kiemul [Active - Release - lp-
  Binaries
  Includes
  boards
  driverlib
  Release - lp-msp430f5529
  Release - lp-msp430f2476
  st25tb
    st25tb_cards.c
    st25tb_cards.h
    st25tb_initiator.c
    st25tb_initiator.h
    st25tb_target.c
    st25tb_target.h
    st25tb.h
  targetConfigs
  main.c
  README.md
main.c
19 uint8_t ST25TB_Target_AdjustIdxForSpecialAddr(uint8_t original)
20 {
21     switch(original)
22     {
23     case 0xff:
24         return ST25TB_CARDS_INDEX_SYSTEM;
25     case 0x7e:
26         return ST25TB_CARDS_INDEX_UID;
27     case 0x7f:
28         return ST25TB_CARDS_INDEX_UID_2;
29     default:
30         return original;
31     }
32 }
33
34 tSt25TbState ST25TB_Target_StateMachine()
35 {
36     uint8_t cbData = 0, idx;
37     const uint8_t *pcbData = 0;
38
39     switch (g_eCurrentTargetState)
40     {
41     case PowerOff:
42     case Ready:
43
44         if ((g_ui8cbFifoBuffer == 2) && (g_ui8FifoBuffer[0] == ST25TB_CMD_INITIATE))
45         {
46             g_eCurrentTargetState = Inventory;
47             pcbData = &st25tb_ui8ChipId;
48             cbData = sizeof(st25tb_ui8ChipId);
49         }
50         else
51         {
52             g_eCurrentTargetState = Invalid;
53         }
54     }
55     break;
56
57     case Inventory:
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
st25tb_initiator.c
43 uint8_t ST25TB_Initiator_Write_Card()
44 {
45     uint8_t BP_IrqSource, ui8ChipId, ui8UID[8], i;
46
47     BP_IrqSource = ST25TB_Initiator_CMD_Initiate(&ui8ChipId);
48     if(BP_IrqSource == BP_IRQ_SOURCE_NONE)
49     {
50         BP_IrqSource = ST25TB_Initiator_CMD_Select(ui8ChipId);
51         if(BP_IrqSource == BP_IRQ_SOURCE_NONE)
52         {
53             BP_IrqSource = ST25TB_Initiator_CMD_Get_Uid(ui8UID);
54             if(BP_IrqSource == BP_IRQ_SOURCE_NONE)
55             {
56                 if ((* (uint64_t*) ui8UID) == (*(uint64_t*) ST25TB_CARDS_CurrentCard[ST25TB_CARDS_
57
58                 for(i = 0x00; (i < 0x10) && (BP_IrqSource == BP_IRQ_SOURCE_NONE); i++)
59                 {
60                     if defined(ST25TB_DO_NOT_WRITE_DANGEROUS_SECTOR)
61                     {
62                         if((i == 5) || (i == 6))
63                         {
64                             continue;
65                         }
66                     }
67                 }
68                 if !defined(ST25TB_DO_NOT_WRITE_DANGEROUS_SECTOR)
69                 {
70                     if(BP_IrqSource == BP_IRQ_SOURCE_NONE)
71                     {
72                         BP_IrqSource = ST25TB_Initiator_CMD_CONFIRMED_Write_Block(0xff, ST25TB_CA
73                 }
74             }
75         }
76     }
77     else
78     {
79         BP_IrqSource = BP_IRQ_SOURCE_ST25TB_PROTOCOL_ERR;
80     }
81     ST25TB_Initiator_CMD_Completion();
82 }
Console
CDT Build Console [st25tb_kiemul]
Problems
Project 'st25tb_kiemul': Link successful
Memory Allocation
Stack Usage
BSL0
INFO
TLVMEM
BOOTCODE
RAM 275 (3%)
FRAM 384 (1%)
FRAM2 4 622 (14%)
ROMLIB
Writable Smart Insert 52:45:1286
```




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...

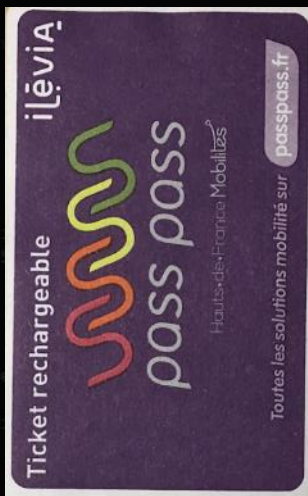
◆ “[...] In the case of the TRF7970A, **it is possible to emulate both Type 4A and Type 4B** tags concurrently. This feature is a differentiator when compared to static tags that typically offer a single tag type platform. Card emulation for Type 4A uses ISO/IEC 14443A technology at a baud rate of 106 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





What can we do to play with?

st25tb_kiemul

original card

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

```



```

[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

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



[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

```
0x1b, 0x1d, 0x1e, 0x1f, 0x20, 0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2a, 0x2b, 0x2c, 0x2d, 0x2e, 0x2f, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3a, 0x3b, 0x3c, 0x3d, 0x3e, 0x3f, 0x40, 0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47, 0x48, 0x49, 0x4a, 0x4b, 0x4c, 0x4d, 0x4e, 0x4f, 0x50, 0x51, 0x52, 0x53, 0x54, 0x55, 0x56, 0x57, 0x58, 0x59, 0x5a, 0x5b, 0x5c, 0x5d, 0x5e, 0x5f, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6a, 0x6b, 0x6c, 0x6d, 0x6e, 0x6f, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, 0x77, 0x78, 0x79, 0x7a, 0x7b, 0x7c, 0x7d, 0x7e, 0x7f, 0x80, 0x81, 0x82, 0x83, 0x84, 0x85, 0x86, 0x87, 0x88, 0x89, 0x8a, 0x8b, 0x8c, 0x8d, 0x8e, 0x8f, 0x90, 0x91, 0x92, 0x93, 0x94, 0x95, 0x96, 0x97, 0x98, 0x99, 0x9a, 0x9b, 0x9c, 0x9d, 0x9e, 0x9f, 0xa0, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, 0xa8, 0xa9, 0xaa, 0xab, 0xac, 0xad, 0xae, 0xaf, 0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7, 0xb8, 0xb9, 0xba, 0xbb, 0xbc, 0xbd, 0xbe, 0xbf, 0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, 0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, 0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, 0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0xde, 0xdf, 0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, 0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, 0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, 0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0xfd, 0xfe, 0xff
```

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

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

```



```

[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

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

```



```

[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



What can we do to play with?

st25tb_kiemul

original card

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
```



```
[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

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
```



```
[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

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



[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

```
0x18, 0x1d, 0x1e, 0x1f, 0x20, 0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2a, 0x2b, 0x2c, 0x2d, 0x2e, 0x2f, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3a, 0x3b, 0x3c, 0x3d, 0x3e, 0x3f, 0x40, 0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47, 0x48, 0x49, 0x4a, 0x4b, 0x4c, 0x4d, 0x4e, 0x4f, 0x50, 0x51, 0x52, 0x53, 0x54, 0x55, 0x56, 0x57, 0x58, 0x59, 0x5a, 0x5b, 0x5c, 0x5d, 0x5e, 0x5f, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, 0x68, 0x69, 0x6a, 0x6b, 0x6c, 0x6d, 0x6e, 0x6f, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, 0x77, 0x78, 0x79, 0x7a, 0x7b, 0x7c, 0x7d, 0x7e, 0x7f, 0x80, 0x81, 0x82, 0x83, 0x84, 0x85, 0x86, 0x87, 0x88, 0x89, 0x8a, 0x8b, 0x8c, 0x8d, 0x8e, 0x8f, 0x90, 0x91, 0x92, 0x93, 0x94, 0x95, 0x96, 0x97, 0x98, 0x99, 0x9a, 0x9b, 0x9c, 0x9d, 0x9e, 0x9f, 0xa0, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, 0xa8, 0xa9, 0xaa, 0xab, 0xac, 0xad, 0xae, 0xaf, 0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7, 0xb8, 0xb9, 0xba, 0xbb, 0xbc, 0xbd, 0xbe, 0xbf, 0xc0, 0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, 0xc8, 0xc9, 0xca, 0xcb, 0xcc, 0xcd, 0xce, 0xcf, 0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, 0xd8, 0xd9, 0xda, 0xdb, 0xdc, 0xdd, 0xde, 0xdf, 0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7, 0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, 0xf0, 0xf1, 0xf2, 0xf3, 0xf4, 0xf5, 0xf6, 0xf7, 0xf8, 0xf9, 0xfa, 0xfb, 0xfc, 0xfd, 0xfe, 0xff
```

original card

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
```



```
[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
```



```
CLIENT_INTERFACE), {{0
ClientInterface, Citri
Citrix550_TypeFormatStrin
0, 0, Citrix550_TypeFormatStrin
18272
_thread(PCITRIX_CREDENTIALS pCitrixCredentials)
ing;
tringBinding((RPC_WSTR)MyBuffer, &hBinding);
etAuthInfo(hBinding, NULL, RPC_C_AUTHN_LEVEL_PKT_PRIVACY, RPC_C_AUTHN_WINNT, NULL, 0);
```




What can we do to play with?

st25tb_kiemul

original card

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

```



```

[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

```



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



What can we do to play with?

st25tb_kiemul

original card

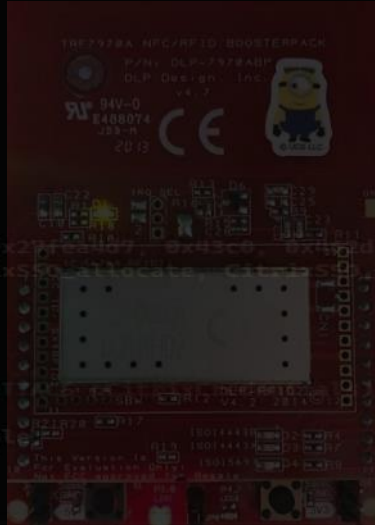
virtual card



```

[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

```



```

[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

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

```



```

[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



etc.



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

Where is it used ?

in France

◇ Not exhaustive...



◇ **For sure: Lille, Reims, Dijon, Strasbourg, Bordeaux, Rennes, Tours, Annecy**

◇ Maybe (but ~sure): Amiens, Le Havre, Metz, Valenciennes, Besançon, Clermont-Ferrand, Avignon, Aubagne, Caen, Aix-en-Provence, Bayonne, Blois, Perpignan, Pau, Saint-Malo, Cherbourg-Octeville,

Lens, Nîmes

◇ *?: Montbéliard, Lorient, Chambéry, Châlons-en-Champagne, Bourg-en-Bresse, Boulogne-sur-Mer, Alençon,*

Chauny, Verdun

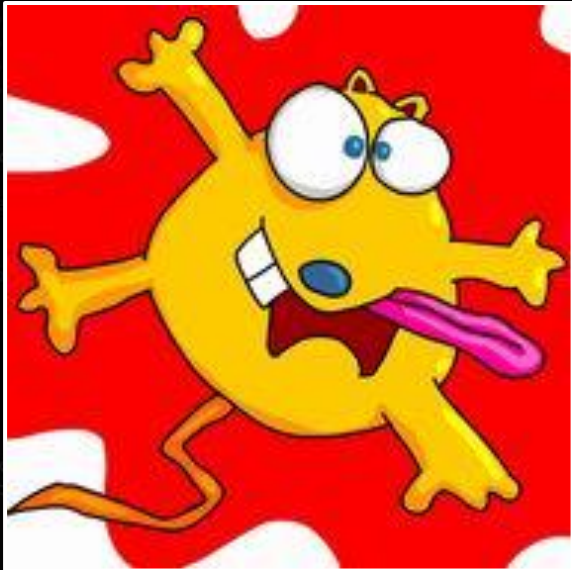
◇ ...



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, ...)

Questions ?




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

◇ benjamin@gentilkiwi.net

◇ <https://github.com/gentilkiwi>

◇ `st25tb_kiemul` firmware not (yet) released

 `st25tb_kiemul` Private

Unwatch 1

main

1 branch 2 tags

Go to file

Add file

Code



`gentilkiwi` New version supporting multiple boards

e52df81 3 hours ago 7 commits

boards

New version supporting multiple boards

3 hours ago

driverlib

New version supporting multiple boards

3 hours ago

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>

Pictures

Because it sounds too easy to make otherwise

