
(/)

Alex Dib

Information Security Enthusiast

(https://twitter.com/scund00r) (https://www.linkedin.com/in/alex-dib-708305108)
(https://github.com/scund00r)

Search

Navigation

» Home (/)

» About Me (/about/)

» Categories (/category/all.html)

» XML Feed (/feed.xml)

Proxmark 3 Cheat Sheet

05 Jun 2018 » all (/category/all.html), rfid (/category/rfid.html)

Table of Contents

Overview

This post will outline commands to read, write, simulate and clone RFID cards using the Proxmark 3 device. Please note this post is a work in progress and will have ongoing updates. These commands were run on Kali using the official and iceman fork Proxmark 3 repo. Commands specific to the iceman fork will be marked with this tag: [Iceman].

The Iceman fork is the most enhanced fork to this day for the Proxmark 3 device. Iceman has done a great job developing and maintaining the repository, please consider donating if you find his fork useful.

13.56MHz

- iClass
- Mifare

125 kHz

- Indala
- HID/ProxCard

Setup

Install

```
# install prerequisites
sudo apt-get install p7zip git build-essential libreadline5 libreadline-dev libusb-0.1-4 li
busb-dev libqt4-dev perl pkg-config wget libncurses5-dev gcc-arm-none-eabi

# check out the latest revision of the official project:
git clone https://github.com/Proxmark/proxmark3.git

# Change directory to the recently cloned Proxmark3 repository
cd proxmark3

# compile the bootrom, OS and software.
make clean && make all
```

Flash

Flash the BOOTROM

```
./client/flasher /dev/ttyACM0 -b bootrom/obj/bootrom.elf
```

Flash the FULLIMAGE

```
./client/flasher /dev/ttyACM0 armsrc/obj/fullimage.elf
```

Run

```
./client/proxmark3 /dev/ttyACM0
```

Commands

Generic

High Frequency search

```
hf search
```

Low Frequency search

```
lf search
```

Measure antenna characteristics, LF/HF voltage should be around 20-45+ V

```
hw tune
```

Check version

```
hw version
```

iClass

iClass Master Key can be found from the following twitter post (https://twitter.com/infosecfriends/status/799003935876870144).

[Iceman] Reverse permute master key

```
# r          reverse permuted key

Example:
hf iclass permute r 3F90EBF0910F7B6F
```

iClass reader

```
hf iclass reader
```

Dump iClass card

```
# k <Key>      : *Access Key as 16 hex symbols or 1 hex to select key from memory

Example:
hf iclass dump k AFA785A7DAB33378
```

Read iClass block

```
# b <Block> : The block number as 2 hex symbols
# k <Key>   : Access Key as 16 hex symbols or 1 hex to select key from memory

hf iclass readblk b 7 k AFA785A7DAB33378
```

Write iClass block

```
# b <Block> : The block number as 2 hex symbols
# d <data>  : Set the Data to write as 16 hex symbols
# k <Key>   : Access Key as 16 hex symbols or 1 hex to select key from memory

hf iclass writeblk b 07 d 6ce099fe7e614fd0 k AFA785A7DAB33378
```

Print keystore

```
# p           : print keys loaded into memory

hf iclass managekeys p
```

Add key to keystore [0-7]

```
# n <keynbr>  : specify the keyNbr to set in memory
# k <key>     : set a key in memory

hf iclass managekeys n 0 k AFA785A7DAB33378
```

Create iclass_decryptionkey.bin

```
echo <auth_key> > key_dump
xxd -r -p key_dump > iclass_decryptionkey.bin
```

Encrypt Block

```
hf iclass encryptblk 0000000f2aa3dba8
```

Load iClass tag dump into memory

```
# f <filename>     : load iclass tag-dump filename

hf iclass eload f iclass_tagdump-db883702f8ff12e0.bin
```

iClass Simulate [0-3]

```
# 0 <CSN> simulate the given CSN
# 1       simulate default CSN
# 2       Reader-attack, gather reader responses to extract elite key
# 3       Full simulation using emulator memory (see 'hf iclass eload')

hf iclass sim 3
```

Simulate iClass card Sequence

```
hf iclass managekeys n 0 k AFA785A7DAB33378
hf iclass dump k 0
hf iclass eload f iclass_tagdump-db883702f8ff12e0.bin
hf iclass sim 3
```

Mifare

Check for default keys

```
# <block number>|<*card memory> <key type (A/B/?)> [t|d|s|ss] [<key (12 hex symbols)>] [<di
c (*.dic)>]
# * - all sectors
# card memory - 0 - MINI(320 bytes), 1 - 1K, 2 - 2K, 4 - 4K, <other> - 1K
# d - write keys to binary file

hf mf chk *1 ? d default_keys.dic
```

Dump Mifare card

```
# [card memory]: 0 = 320 bytes (Mifare Mini), 1 = 1K (default), 2 = 2K, 4 = 4K

hf mf dump 1
```

Convert .bin to .eml

```
script run dumptoemul -i dumpdata.bin
```

Read Mifare block

```
# b <no>  : block to read
# k <key> : (optional) key for authentication

hf mf rdbl b 3 k FFFFFFFF
```

Write Mifare block

```
# <block number> <key A/B> <key (12 hex symbols)> <block data (32 hex symbols)>

hf mf wrbl 0 A FFFFFFFFFFFF d3a2859f6b880400c801002000000016
```

Hardnested attack

```
# <block number> <key A|B> <key (12 hex symbols)>
# <target block number> <target key A|B> [known target key (12 hex symbols)] [w] [s]
# w: Acquire nonces and write them to binary file nonces.bin

hf mf hardnested 0 A 8829da9daf76 4 A w
```

Load Mifare tag dump into memory

```
hf mf eload 353C2AA6
```

Mifare Simulate [0-3]

```
# u    (Optional) UID 4,7 or 10 bytes. If not specified, the UID 4B from emulator memory wi
ll be used

hf mf sim u 353c2aa6
```

Simulate Mifare card Sequence

```
hf mf chk *1 ? d default_keys.dic
hf mf dump 1
script run dumptoemul -i dumpdata.bin
hf mf eload 353C2AA6
hf mf sim u 353c2aa6
```

Indala

Read Indala card

```
lf indala read
```

Demodulate Indala card

```
lf indala demod
```

[Iceman] Simulate Indala card

```
# <uid> :  64/224 UID

lf indala sim a0000000c2c436c1
```

Clone to T55x7 card

```
# <uid> :  64/224 UID

lf indala clone a0000000c2c436c1
```

HID/ProxCard

Read ProxCard card

```
lf hid read
```

Demodulate ProxCard card

```
lf hid demod
```

[Iceman] Convert Facility code & Card number to Wiegand

```
# [OEM] [FC] [CN]
# OEM           - OEM number / site code
# FC            - facility code
# CN            - card number

lf hid wiegand 0 56 150
```

Simulate card

```
# <ID>

lf hid sim 200670012d
```

Clone to T55x7 card

```
# <ID>

lf hid clone 200670012d
```

T55xx

Detect card

```
lf t55xx detect
```

Set demodulation

```
# d <FSK|FSK1|FSK1a|FSK2|FSK2a|ASK|PSK1|PSK2|NRZ|BI|BIa>  Set demodulation FSK / ASK / PSK
/ NRZ / Biphase / Biphase A
# EM is ASK
# HID Prox is FSK
# Indala is PSK

lf t55xx config FSK
```

Write T55xx block

```
# b <block>    - block number to write. Between 0-7
# d <data>     - 4 bytes of data to write (8 hex characters)

lf t55xx wr b 0 d 00081040
```

Wipe a T55xx tag and set defaults

```
lf t55xx wipe
```

Data

Get raw samples [512-40000]

```
data samples <size>
```

Save to file

```
data save <filename>
```

Load from file

```
data load <filename>
```

Lua Scripts

List Lua Scripts

```
script list
```

Convert .bin to .eml

```
# i <file>              Specifies the dump-file (input). If omitted, 'dumpdata.bin' is used

script run dumptoemul -i xxxxxxxxxxxxx.bin
```

Format Mifare card

```
# k <key>       - the current six byte key with write access
# n <key>       - the new key that will be written to the card
# a <access>    - the new access bytes that will be written to the card
# x             - execute the commands aswell.

script run formatMifare -k FFFFFFFFFFFF -n FFFFFFFFFFFF -x
```

Links

- Official Proxmark 3 (https://github.com/Proxmark/proxmark3)
- Iceman fork (https://github.com/iceman1001/proxmark3)
- 0xFFFF's Cardinfo Tool (http://cardinfo.barkweb.com.au/index.php)
- Smart Card Wiki (http://smartcard.wiki/start)
- Iceman website (http://www.icedev.se/pm3.aspx)
- Proxmark Forums (http://www.proxmark.org/forum/index.php)
- Dumping iClass Keys (http://blog.opensecurityresearch.com/2012/11/dumping-iclass-keys.html)
- Reverse Engineering iClass Keys (https://blog.kchung.co/reverse-engineering-hid-iclass-master-keys/)
- Heart of Darkness (https://www.openpcd.org/dl/HID-iCLASS-security.pdf)

Related Posts

- Red Team & Physical Entry Gear (https://scund00r.com/all/gear/2019/06/25/red-team-and-physical-entry-gear.html) (Categories: all (/category/all.html), gear (/category/gear.html))
- RFID Thief v2.0 (https://scund00r.com/all/rfid/tutorial/2018/07/12/rfid-theif-v2.html) (Categories: all (/category/all.html), rfid (/category/rfid.html), tutorial (/category/tutorial.html))
- Debricking Proxmark 3 using the Bus Pirate (https://scund00r.com/all/rfid/2018/05/18/debrick-proxmark.html) (Categories: all (/category/all.html), rfid (/category/rfid.html))
- Passing OSCP (https://scund00r.com/all/oscp/2018/02/25/passing-oscp.html) (Categories: all (/category/all.html), oscp (/category/oscp.html))

« Debricking Proxmark 3 using the Bus Pirate (/all/rfid/2018/05/18/debrick-proxmark.html)

RFID Thief v2.0 » (/all/rfid/tutorial/2018/07/12/rfid-theif-v2.html)