Command Cheat Sheet

Generic	Low Frequency 125 kHz	High Frequency 13.56 MHz
Generic	T55XX	MIFARE
Data	HID Prox	iCLASS
Memory	Indala	
Sim Module	Hitag	
Lua Scripts		
Smart Card		
Wiegand convertion		

Generic

^Top

Identify High Frequency cards

```
pm3 --> hf search
```

Identify Low Frequency cards

```
pm3 --> lf search
```

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

```
pm3 --> hw tune
```

Check versioning

```
pm3 --> hw version
```

Check overall status

```
pm3 --> hw status
```

iCLASS

^Top

Reverse permute iCLASS master key

```
Options
---
-r  --reverse      : reverse permuted key
    --key <bytes>  : input key

pm3 --> hf iclass permute --reverse --key 3F90EBF0910F7B6F
```

iCLASS Reader

```
pm3 --> hf iclass reader
```

Dump iCLASS card contents

```
Options
---
-f, --file <filename>          filename to save dump to
-k, --key <hex>                debit key as 16 hex symbols OR
NR/MAC for replay
    --ki <dec>                 debit key index to select key
from memory 'hf iclass managekeys'
    --credit <hex>             credit key as 16 hex symbols
    --ci <dec>                 credit key index to select key
from memory 'hf iclass managekeys'
    --elite                    elite computations applied to
key
    --raw                      raw, the key is interpreted as
raw block 3/4
    --nr                       replay of NR/MAC

pm3 --> hf iclass dump --ki 0
```

Read iCLASS Block

```
Options
---
-k, --key <hex>                Access key as 16 hex symbols
-b, --block <dec>              The block number to read as an
```

```
integer
    --ki <dec>                  Key index to select key from
memory 'hf iclass managekeys'
    --credit                    key is assumed to be the
credit key
    --elite                     elite computations applied to
key
    --raw                       no computations applied to key
(raw)
    --nr                        replay of NR/MAC

pm3 --> hf iclass rdbl -b 7 --ki 0
```

Write to iCLASS Block

```
Options
---
-k, --key <hex>                 Access key as 16 hex symbols
-b, --block <dec>               The block number to read as an
integer
-d, --data <hex>                data to write as 16 hex
symbols
    --ki <dec>                  Key index to select key from
memory 'hf iclass managekeys'
    --credit                    key is assumed to be the
credit key
    --elite                     elite computations applied to
key
    --raw                       no computations applied to key
(raw)
    --nr                        replay of NR/MAC

pm3 --> hf iclass wrbl -b 7 -d 6ce099fe7e614fd0 --ki 0
```

Print keystore

```
Options
---
-p, --print                     Print keys loaded into memory
```

```
pm3 --> hf iclass managekeys -p
```

Add key to keystore [0-7]

```
Options
---
-f, --file <filename>          Specify a filename to use with
load or save operations
    --ki <dec>                 Specify key index to set key
in memory

pm3 --> hf iclass managekeys --ki 3 -k AFA785A7DAB33378
```

Encrypt iCLASS Block

```
Options
---
-d, --data <hex>               data to encrypt
-k, --key <hex>                3DES transport key
-v, --verbose                  verbose output

pm3 --> hf iclass encrypt -d 0000000f2aa3dba8
```

Decrypt iCLASS Block / file

```
Options
---
-f, --file <filename>          filename of dumpfile
-d, --data <hex>               3DES encrypted data
-k, --key <hex>                3DES transport key
-v, --verbose                  verbose output

pm3 --> hf iclass decrypt -d 2AD4C8211F996871
pm3 --> hf iclass decrypt -f hf-iclass-db883702f8ff12e0.bin
```

Load iCLASS dump into memory for simulation

```
Options
---
-f, --file <filename>          filename of dump
    --json                     load JSON type dump
    --eml                      load EML type dump

pm3 --> hf iclass eload -f hf-iclass-db883702f8ff12e0.bin
```

Clone iCLASS Legacy Sequence

```
pm3 --> hf iclass rdbl -b 7 --ki 0
pm3 --> hf iclass wrbl -b 7 -d 6ce099fe7e614fd0 --ki 0
```

Simulate iCLASS

```
Options
---
-t, --type <int>               Simulation type to use
    --csn <hex>                Specify CSN as 8 bytes (16 hex
symbols) to use with sim type 0
Types:
    0    simulate the given CSN
    1    simulate default CSN
    2    runs online part of LOCLASS attack
    3    full simulation using emulator memory (see 'hf
iclass eload')
    4    runs online part of LOCLASS attack against reader in
keyroll mode

pm3 --> hf iclass sim -t 3
```

Simulate iCLASS Sequence

```
pm3 --> hf iclass dump --ki 0
pm3 --> hf iclass eload -f hf-iclass-db883702f8ff12e0.bin
```

```
pm3 --> hf iclass sim -t 3
```

Extract custom iCLASS key (loclass attack)

```
Options
---
-f <filename>                  specify a filename to clone
from
-k <key>                       Access Key as 16 hex symbols
or 1 hex to select key from memory
    --elite                    Elite computations applied to
key

pm3 --> hf iclass sim -t 2
pm3 --> hf iclass loclass -f iclass_mac_attack.bin
pm3 --> hf iclass managekeys --ki 7 -k <Kcus>
pm3 --> hf iclass dump --ki 7 --elite
```

Verify custom iCLASS key

```
options
---
-f, --file <filename>          Dictionary file with default
iclass keys
    --csn <hex>                Specify CSN as 8 bytes (16 hex
symbols)
    --epurse <hex>             Specify ePurse as 8 bytes (16
hex symbols)
    --macs <hex>               MACs
    --raw                      no computations applied to key
(raw)
    --elite                    Elite computations applied to
key

pm3 --> hf iclass lookup --csn 010a0ffff7ff12e0 --epurse
feffffffffffffff --macs 66348979153c41b9 -f
iclass_default_keys --elite
```

MIFARE

^Top

Check for default keys

```
options
---
-k, --key <hex>                Key specified as 12 hex
symbols
    --blk <dec>                Input block number
-a                             Target Key A, if found also
check Key B for duplicate
-b                             Target Key B
-*, --all                      Target both key A & B
(default)
    --mini                     MIFARE Classic Mini / S20
    --1k                       MIFARE Classic 1k / S50
(default)
    --2k                       MIFARE Classic/Plus 2k
    --4k                       MIFARE Classic 4k / S70
    --emu                      Fill simulator keys from found
keys
    --dump                     Dump found keys to binary file
-f, --file <filename>          filename of dictionary

pm3 --> hf mf chk --1k -f mfc_default_keys
```

Check for default keys from local memory

```
Options
---
-k, --key <hex>                Key specified as 12 hex
symbols
    --mini                     MIFARE Classic Mini / S20
    --1k                       MIFARE Classic 1k / S50
(default)
    --2k                       MIFARE Classic/Plus 2k
    --4k                       MIFARE Classic 4k / S70
```

```
    --emu                       Fill simulator keys from found
keys
    --dump                      Dump found keys to binary file
    --mem                       Use dictionary from
flashmemory
-f, --file <filename>           filename of dictionary

pm3 --> hf mf fchk --1k --mem
```

Dump MIFARE Classic card contents

```
Options:
---
-f, --file <filename>           filename of dump
-k, --keys <filename>           filename of keys
    --mini                      MIFARE Classic Mini / S20
    --1k                        MIFARE Classic 1k / S50
(default)
    --2k                        MIFARE Classic/Plus 2k
    --4k                        MIFARE Classic 4k / S70

pm3 --> hf mf dump
pm3 --> hf mf dump --1k -k hf-mf-A29558E4-key.bin -f hf-mf-
A29558E4-dump.bin
```

Write to MIFARE Classic block

```
Options:
---
    --blk <dec>                 block number
-a                              input key type is key A (def)
-b                              input key type is key B
-k, --key <hex>                 key, 6 hex bytes
-d, --data <hex>                bytes to write, 16 hex bytes

pm3 --> hf mf wrbl --blk 0 -k FFFFFFFFFFFF -d
d3a2859f6b880400c801002000000016
```

Run autopwn, to extract all keys and backup a MIFARE Classic tag

```
Options:
---
-k, --key <hex>                Known key, 12 hex bytes
-s, --sector <dec>             Input sector number
-a                             Input key A (def)
-b                             Input key B
-f, --file <fn>                filename of dictionary
-s, --slow                     Slower acquisition (required
by some non standard cards)
-l, --legacy                   legacy mode (use the slow `hf
mf chk`)
-v, --verbose                  verbose output (statistics)
    --mini                     MIFARE Classic Mini / S20
    --1k                       MIFARE Classic 1k / S50
(default)
    --2k                       MIFARE Classic/Plus 2k
    --4k                       MIFARE Classic 4k / S70

pm3 --> hf mf autopwn

// target MFC 1K card, Sector 0 with known key A
'FFFFFFFFFFFF'
pm3 --> hf mf autopwn -s 0 -a -k FFFFFFFFFFFF

// target MFC 1K card, default dictionary
pm3 --> hf mf autopwn --1k -f mfc_default_keys
```

Run hardnested attack

```
Options
---
-k, --key <hex>                Key, 12 hex bytes
    --blk <dec>                Input block number
-a                             Input key A (def)
-b                             Input key B
    --tblk <dec>               Target block number
    --ta                       Target key A
    --tb                       Target key B
```

```
    --tk <hex>                  Target key, 12 hex bytes
-f, --file <fn>                 R/W <name> instead of default
name
-s, --slow                      Slower acquisition (required
by some non standard cards)
-w, --wr                        Acquire nonces and UID, and
write them to file `hf-mf-<UID>-nonces.bin`

pm3 --> hf mf hardnested --blk 0 -a -k 8829da9daf76 --tblk 4
--ta -w
```

Load MIFARE Classic dump file into emulator memory for simulation Accepts (BIN/EML/JSON)

```
Options
---
-f, --file <fn>                 filename of dump
    --mini                      MIFARE Classic Mini / S20
    --1k                        MIFARE Classic 1k / S50 (def)
    --2k                        MIFARE Classic/Plus 2k
    --4k                        MIFARE Classic 4k / S70
    --ul                        MIFARE Ultralight family
-q, --qty <dec>                 manually set number of blocks
(overrides)

pm3 --> hf mf eload -f hf-mf-353C2AA6-dump.bin
pm3 --> hf mf eload --1k -f hf-mf-353C2AA6-dump.bin
```

Simulate MIFARE

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

pm3 --> hf mf sim -u 353c2aa6
```

Simulate MIFARE Sequence

```
pm3 --> hf mf fchk --1k -f mfc_default_keys.dic
pm3 --> hf mf dump
pm3 --> hf mf eload -f hf-mf-<UID>-dump.bin
pm3 --> hf mf sim -u 353c2aa6
```

Clone MIFARE 1K Sequence

```
pm3 --> hf mf fchk --1k -f mfc_default_keys.dic
pm3 --> hf mf dump
pm3 --> hf mf restore --1k --uid 4A6CE843 -k hf-mf-A29558E4-
key.bin -f hf-mf-A29558E4-dump.bin
```

Read MIFARE Ultralight EV1

```
pm3 --> hf mfu info
```

Clone MIFARE Ultralight EV1 Sequence

```
pm3 --> hf mfu dump -k FFFFFFFF
pm3 --> hf mfu eload -f hf-mfu-XXXX-dump.bin
pm3 --> hf mfu sim -t 7
```

Bruteforce MIFARE Classic card numbers from 11223344 to 11223346

```
pm3 --> script run hf_mf_uidbruteforce -s 0x11223344 -e
0x11223346 -t 1000 -x mfc
```

Bruteforce MIFARE Ultralight EV1 card numbers from 11223344556677 to 11223344556679

```
pm3 --> script run hf_mf_uidbruteforce -s 0x11223344556677 -e
0x11223344556679 -t 1000 -x mfu
```

Wiegand manipulation

^Top

List all available wiegand formats in client

```
pm3 --> wiegand list
```

Convert Site & Facility code to Wiegand raw hex

```
Options
---
    --fc <dec>                  facility number
    --cn <dec>                  card number
    --issue <dec>               issue level
    --oem <dec>                 OEM code
-w, --wiegand <format>          see `wiegand list` for
available formats
    --pre                       add HID ProxII preamble to
wiegand output

pm3 --> wiegand encode -w H10301 --oem 0 --fc 101 --cn 1337
pm3 --> wiegand encode --fc 101 --cn 1337
```

Convert Site & Facility code from Wiegand raw hex to numbers

```
Options
---
-p, --parity                    ignore invalid parity
-r, --raw <hex>                 raw hex to be decoded
-b, --bin <bin>                 binary string to be decoded

pm3 --> wiegand decode --raw 2006f623ae
```

HID Prox

^Top

Read HID Prox card

```
pm3 --> lf hid read
```

Demodulate HID Prox card

```
pm3 --> lf hid demod
```

Simulate Prox card

```
pm3 --> lf hid sim -r 200670012d
pm3 --> lf hid sim -w H10301 --fc 10 --cn 1337
```

Clone Prox to T5577 card

```
pm3 --> lf hid clone -r 200670012d
pm3 --> lf hid clone -w H10301 --fc 10 --cn 1337
```

Brute force HID reader

```
Options
---
-v, --verbose                   verbose logging, show all
tries
-w, --wiegand format            see `wiegand list` for
available formats
-f, --fn dec                    facility code
-c, --cn dec                    card number to start with
-i dec                          issue level
-o, --oem dec                   OEM code
-d, --delay dec                 delay betweens attempts in ms.
Default 1000ms
```

```
    --up                        direction to increment card
number. (default is both directions)
    --down                      direction to decrement card
number. (default is both directions)

pm3 --> lf hid brute -w H10301 -f 224
pm3 --> lf hid brute -v -w H10301 -f 21 -c 200 -d 2000
```

Indala

^Top

Read Indala card

```
pm3 --> lf indala read
```

Demodulate Indala card

```
pm3 --> lf indala demod
```

Simulate Indala card

```
Options
---
-r, --raw <hex>                 raw bytes
    --heden <decimal>           Cardnumber for Heden 2L format

pm3 --> lf indala sim -r a0000000c2c436c1
```

Clone to T55x7 card

```
Options
---
-r, --raw <hex>                 raw bytes
```

```
    --heden <decimal>          Cardnumber for Heden 2L format
    --fc <decimal>             Facility Code (26 bit H10301
format)
    --cn <decimal>             Cardnumber (26 bit H10301
format)
    --q5                       specify writing to Q5/T5555
tag
    --em                       specify writing to EM4305/4469
tag

pm3 --> lf indala clone -r a0000000c2c436c1
```

Hitag

^Top

Read Hitag information

```
pm3 --> lf hitag info
```

Act as Hitag reader

```
Options
---
    --01                       HitagS, read all pages,
challenge mode
    --02                       HitagS, read all pages, crypto
mode. Set key=0 for no auth

    --21                       Hitag2, read all pages,
password mode. def 4D494B52 (MIKR)
    --22                       Hitag2, read all pages,
challenge mode
    --23                       Hitag2, read all pages, crypto
mode. Key ISK high + ISK low. def 4F4E4D494B52 (ONMIKR)
    --25                       Hitag2, test recorded
authentications (replay?)
```

```
    --26                        Hitag2, read UID
-k, --key <hex>                 key, 4 or 6 hex bytes
    --nrar <hex>                nonce / answer reader, 8 hex
bytes

pm3 --> lf hitag --26
pm3 --> lf hitag --21 -k 4D494B52
pm3 --> lf hitag reader --23 -k 4F4E4D494B52
```

Sniff Hitag traffic

```
pm3 --> lf hitag sniff
pm3 --> lf hitag list
```

Simulate Hitag2

```
pm3 --> lf hitag sim -2
```

Write to Hitag block

```
Options
---
    --03                        HitagS, write page, challenge
mode
    --04                        HitagS, write page, crypto
mode. Set key=0 for no auth

    --24                        Hitag2, write page, crypto
mode.
    --27                        Hitag2, write page, password
mode
-p, --page <dec>                page address to write to
-d, --data <hex>                data, 4 hex bytes
-k, --key <hex>                 key, 4 or 6 hex bytes
    --nrar <hex>                nonce / answer writer, 8 hex
bytes

pm3 --> lf hitag writer --24 -k 499602D2 -p 1 -d 00000000
```

Simulate Hitag2 sequence

```
pm3 --> lf hitag reader --21 -k 56713368
pm3 --> lf hitag sim -2
```

T55XX

^Top

Detect T55XX card

```
pm3 --> lf t55xx detect
```

Configure modulation

```
Options
---
    --FSK                       set demodulation FSK
    --FSK1                      set demodulation FSK 1
    --FSK1A                     set demodulation FSK 1a (inv)
    --FSK2                      set demodulation FSK 2
    --FSK2A                     set demodulation FSK 2a (inv)
    --ASK                       set demodulation ASK
    --PSK1                      set demodulation PSK 1
    --PSK2                      set demodulation PSK 2
    --PSK3                      set demodulation PSK 3
    --NRZ                       set demodulation NRZ
    --BI                        set demodulation Biphase
    --BIA                       set demodulation Diphase
(inverted biphase)

EM is ASK
HID Prox is FSK
Indala is PSK
```

```
pm3 --> lf t55xx config --FSK
```

Set timings to default

```
Options
---

-p, --persist                  persist to flash memory (RDV4)
-z                             Set default t55x7 timings (use
`-p` to save if required)
pm3 --> lf t55xx deviceconfig -zp
```

Write to T55xx block

```
-b, --blk <0-7>                block number to write
-d, --data <hex>               data to write (4 hex bytes)
-p, --pwd <hex>                password (4 hex bytes)

pm3 --> lf t55xx write -b 0 -d 00081040
```

Wipe a T55xx tag and set defaults

```
pm3 --> lf t55xx wipe
```

Data

^Top

Get raw samples [512-40000]

```
pm3 --> data samples -n <size>
```

Save samples to file

```
pm3 --> data save -f <filename>
```

Load samples from file

```
pm3 --> data load -f <filename>
```

769 lines (634 sloc) | 22.7 KB

Raw Blame

^Top

List lua Scripts

```
pm3 --> script list
```

View lua helptext

```
pm3 --> script run <nameofscript> -h
```

Convert .bin to .eml

```
Options
---
-i <file>                       Specifies the dump-file
(input). If omitted, 'dumpdata.bin' is used
-o <filename>                   Specifies the output file. If
omitted, <uid>.eml is used

pm3 --> script run data_mf_bin2eml -i xxxxxxxxxxxxxx.bin
```

Convert .eml to .bin

```
Options
```

```
---
-i <filename>                   Specifies the dump-file
(input). If omitted, 'dumpdata.eml' is used
-o <filename>                   Specifies the output file. If
omitted, <currdate>.bin is used

pm3 --> script run data_mf_eml2bin -i myfile.eml -o
myfile.bin
```

Format Mifare card

```
Options
---
-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 as well

pm3 --> script run hf_mf_format -k FFFFFFFFFFFF -n
FFFFFFFFFFFF -x
```

Memory

^Top

Load default keys into flash memory (RDV4 only)

```
Options
---
-o <offset>                     offset in memory
-f <filename>                   file name
    --mfc                       upload 6 bytes keys (mifare
key dictionary)
    --iclass                    upload 8 bytes keys (iClass
```

```
key dictionary)
    --t55xx                    upload 4 bytes keys (pwd
dictionary)

pm3 --> mem load -f mfc_default_keys --mfc
pm3 --> mem load -f t55xx_default_pwds --t5xx
pm3 --> mem load -f iclass_default_keys --iclass
```

Sim Module

^Top

Upgrade Sim Module firmware

```
pm3 --> smart upgrade -f sim013.bin
```

Smart Card

^Top

Get Smart Card Information

```
pm3 --> smart info
```

Act like an IS07816 reader

```
pm3 --> smart reader
```

Set clock speed for smart card interface

```
Options
---
```

```
    --16mhz                     16 MHz clock speed
    --8mhz                      8 MHz clock speed
    --4mhz                      4 MHz clock speed


pm3 --> smart setclock --8mhz
```

Send raw hex data

```
Options
---
-r                              do not read response
-a                              active smartcard without
select (reset sc module)
-s                              active smartcard with select
(get ATR)
-t, --tlv                       executes TLV decoder if it
possible
-0                              use protocol T=0
-d, --data <hex>                bytes to send

pm3 --> smart raw -s -0 -d
00a404000e315041592e5359532e4444463031
pm3 --> smart raw -0 -d
00a404000e325041592e5359532e4444463031
pm3 --> smart raw -0 -t -d 00a4040007a0000000041010
pm3 --> smart raw -0 -t -d 00a4040007a0000000031010
```

Bruteforce SPI

```
Options
---
-t, --tlv                       executes TLV decoder if it
possible

pm3 --> smart brute
pm3 --> smart brute --tlv
```