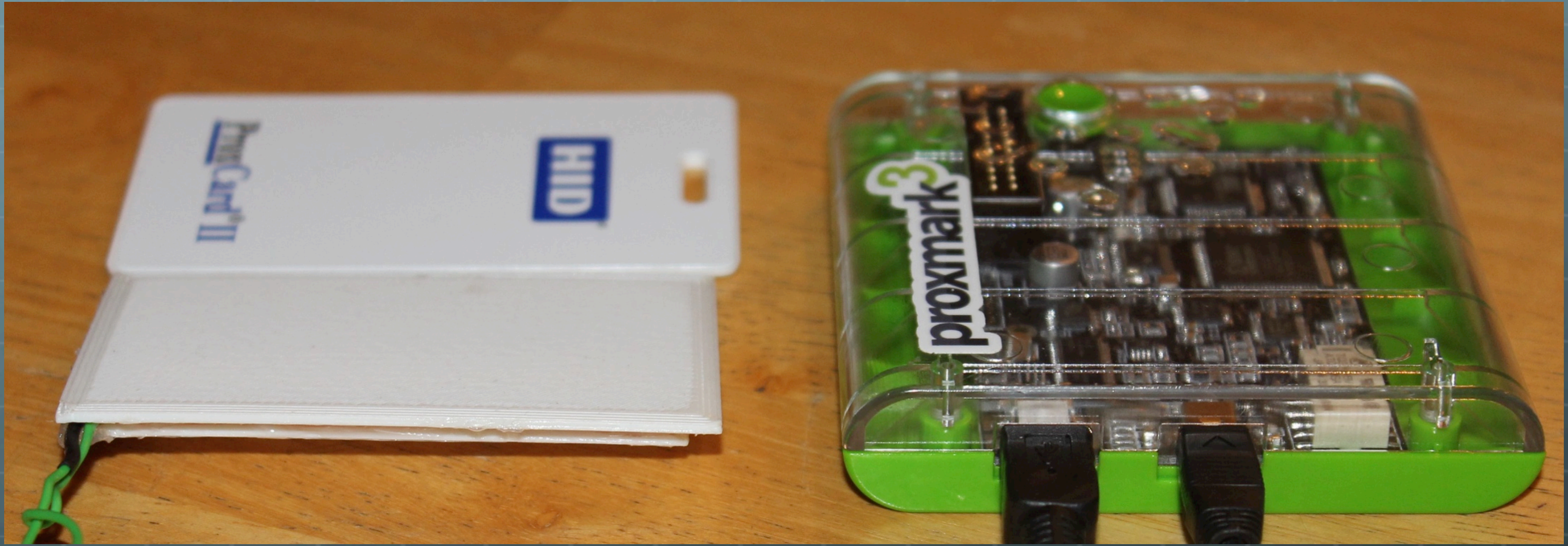


How to Train Your RFID Tools

Craig Young
Security Researcher @ Tripwire VERT

DEF CON 23 LAS VEGAS

About Craig

- Security Researcher in Tripwire VERT
  - Author of IP360 content and original research
  - Identified 100+ CVEs over 2013 & 2014
  - Won 1st SOHOpelessly Broken @ DC22 using 10 0-days

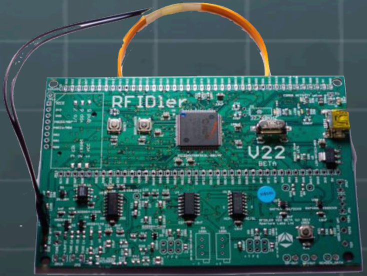
Motivation/Goals

- Shed light on available RFID hacking tools
  - Document hardware/software layouts, use cases, etc.
  - Release new features for an open source tool
  - Experiment with paired embedded systems
- Explore practical attacks on RFID systems
  - LF (125-134kHz): access control badges, pet IDs, etc
  - HF (13.56MHz): passports, payment systems, phones

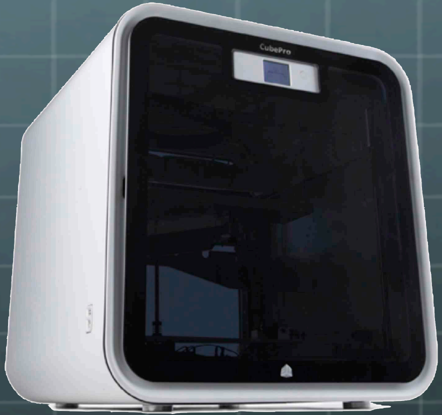
Tools

Proxmark3

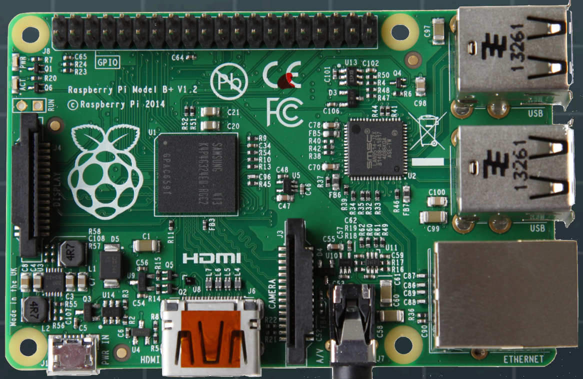
NFC USB (PN533)



RFIDler v22


CubePro 3D Printer

USB Armory



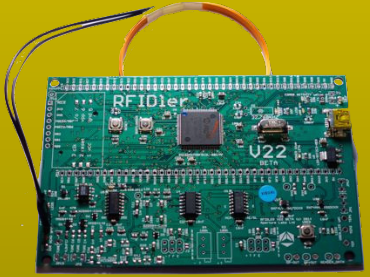
Raspberry Pi

Tools

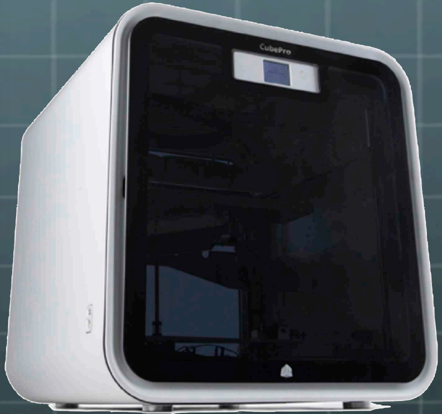
RFID

Proxmark3

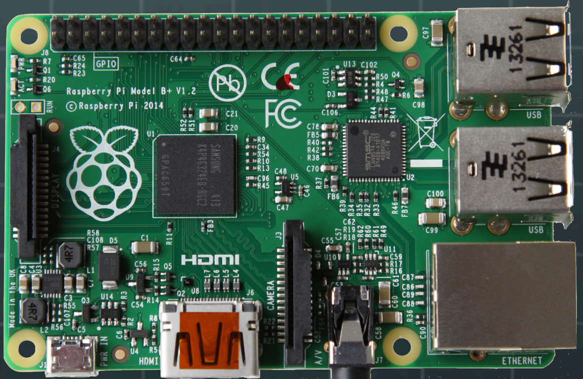
NFC USB (PN533)



RFIDler v22


CubePro 3D Printer

USB Armory



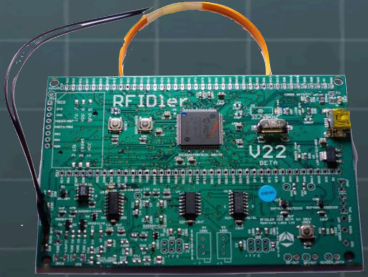
Raspberry Pi

Tools

Fabrication

Proxmark3

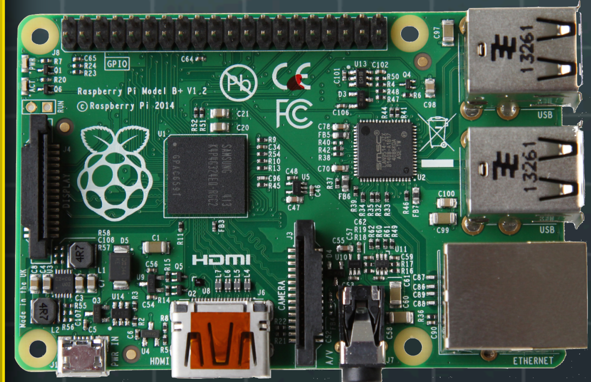
NFC USB (PN533)



RFIDler v22

CubePro 3D Printer

USB Armory

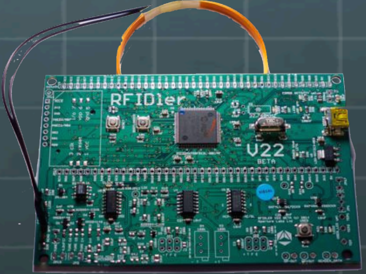


Raspberry Pi

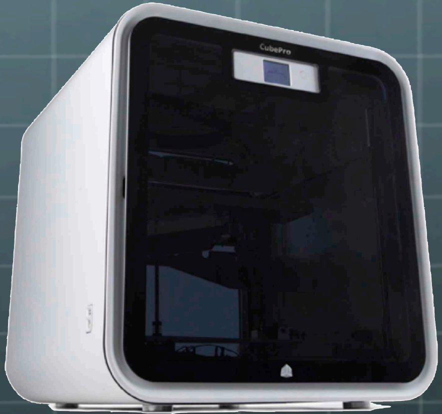
Tools

Proxmark3

NFC USB (PN533)



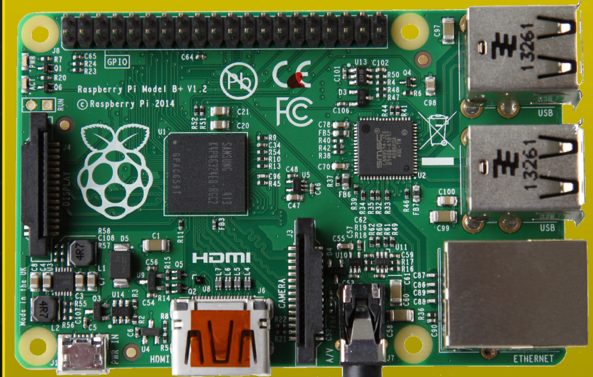
RFIDler v22



CubePro 3D Printer

Computing

USB Armory



Raspberry Pi

Objectives

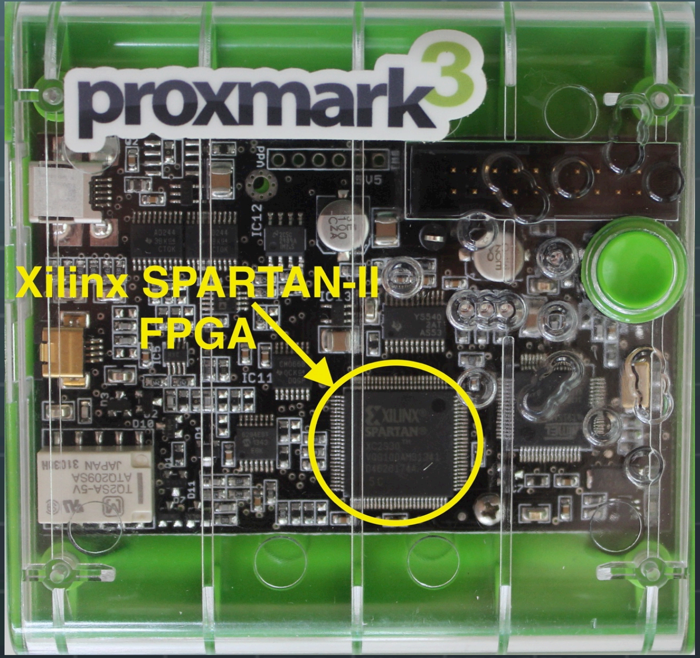
- Decipher RFID hacking hardware for others
  - Understanding the hardware
  - Making and documenting firmware patches
- Investigate practical RFID attacks
  - Tag basics: reading, cloning, and emulation
  - Using 3D printing to assist in attacks
  - Hiding host devices for complex actions

Proxmark3 Intro

- Adopted from Gerhard de Koning Gans' thesis
  - Koning Gans was analyzing Mifare transit cards
  - This system combines cheap FPGA and ARM CPU
  - Completely open source HW and SW
  - Low & High Frequency support

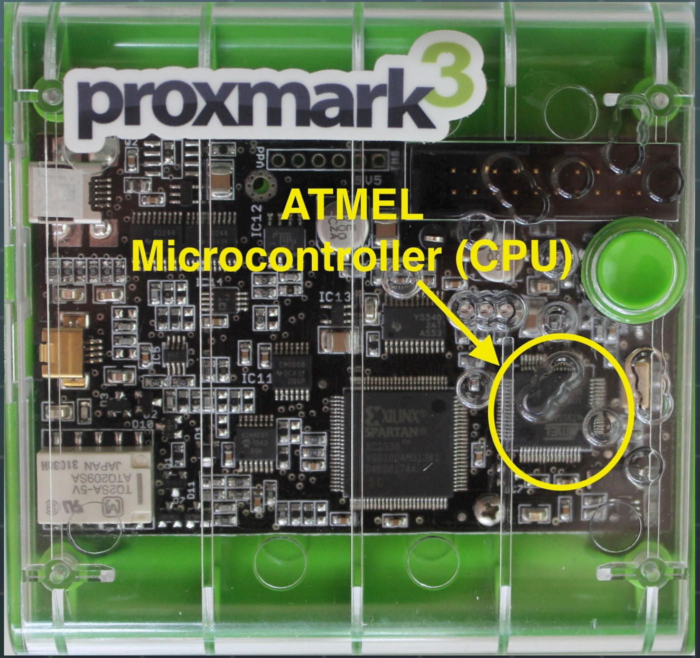
The Proxmark3 FPGA

- Xilinx Spartan-II Field-Programmable Gate Array
  - Written in Verilog for ISE WebPACK
  - Serial Peripheral Interface (SPI)
    - Coil driver
    - Analogue/Digital Converters
  - Synchronous Serial Port to CPU
  - Module based design
  - Modes specify module connections



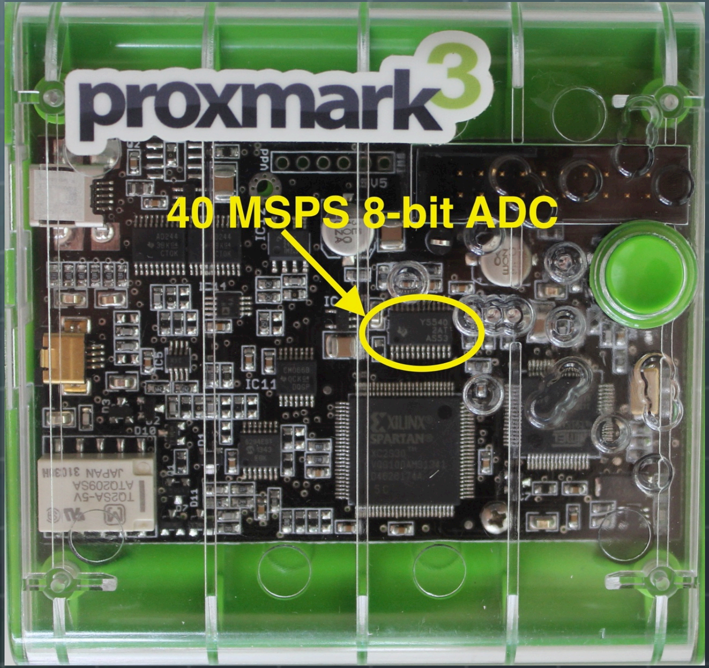
Proxmark3 CPU

- ATMEL 32-bit RISC Processor (AT91SAM7Sxx Series)
  - Coded in C
  - Receives data from FPGA
  - Encoding/decoding of bitstream
  - Handle USB data link
  - All high level functions run here



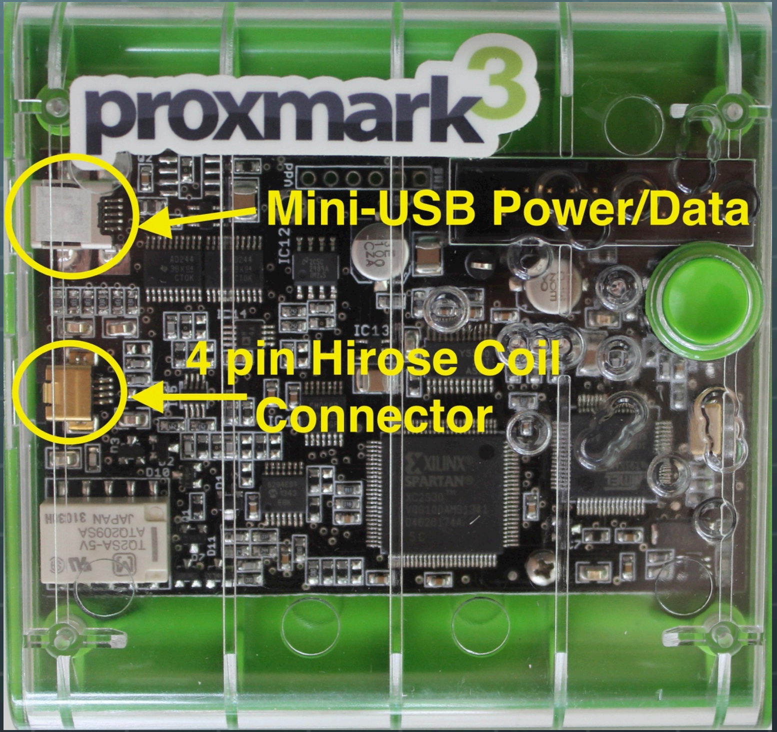
Proxmark3 ADC

- TI analog-to-digital converter (TLC5540)
  - 8-bit resolution
  - 40 Megasamples/sec (MSPS)
  - Analog input from coil
  - 8-pin output to FPGA
  - Control via SPI



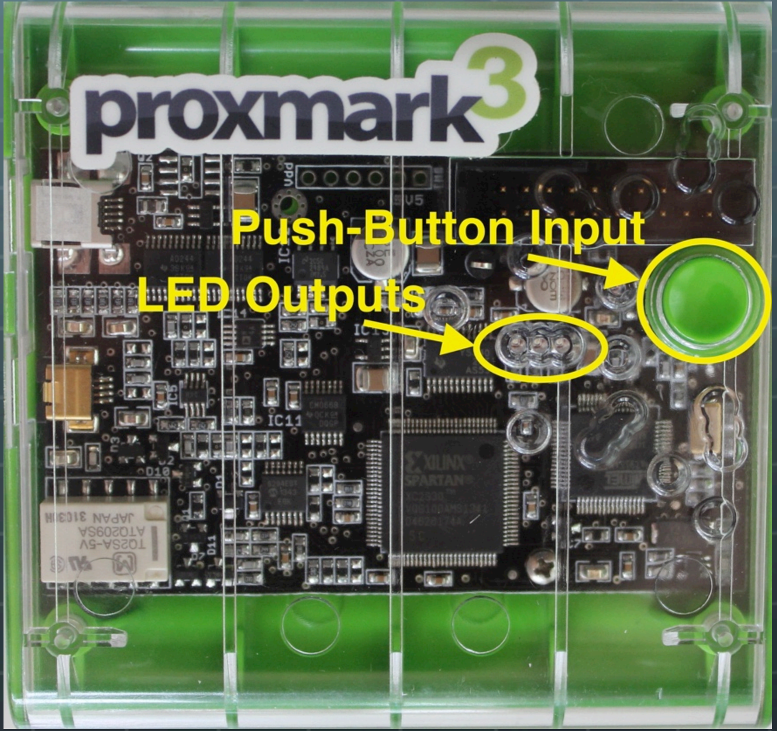
Proxmark3 Connectors

- Mini-USB
  - Provides Power
  - Data connection via USB-Serial
- Hirose Connector
  - Connection to antenna coil(s)
  - 2 wires for LF and 2 for HF



Proxmark3 Human I/O

- Single push button
  - Control PM3 without serial
  - Terminate active operations
  - Initiate firmware update
- Triple LED color coded display
  - Indicate boot mode
  - Stand-alone status lights

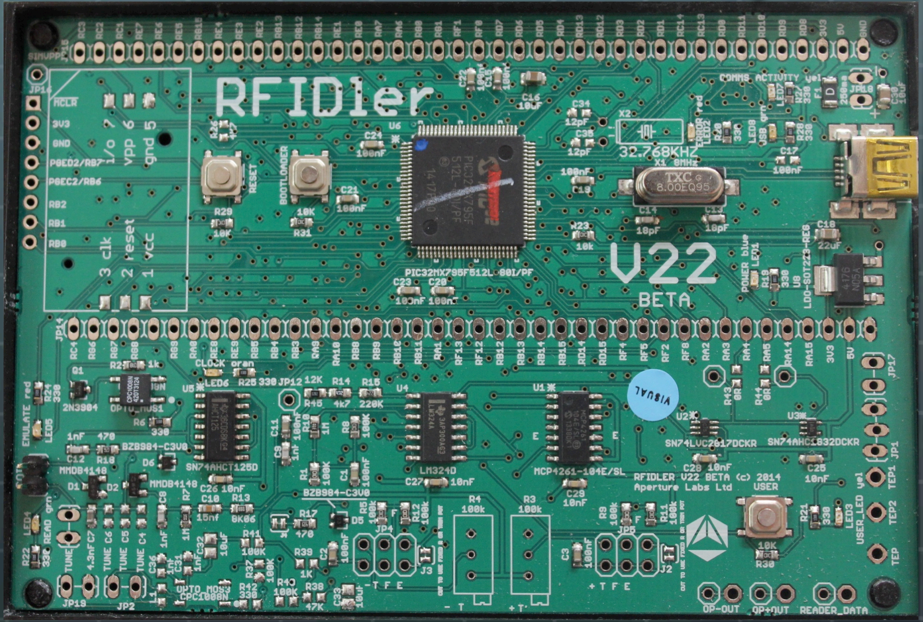


PN533 NFC USB

- NXP Chipset for Near Field Communications
- Hardware implementation of ISO 14443 standards
  - MIFARE Family (ISO/IEC 14443A)
  - NFC Forum Type 4 Tag 2.0 (ISO/IEC 14443B)
  - Sony Felicity Tags (FeliCa reader/writer)
  - Peer-to-Peer Data Transfer (ISO/IEC 18092)
- libNFC Compatible

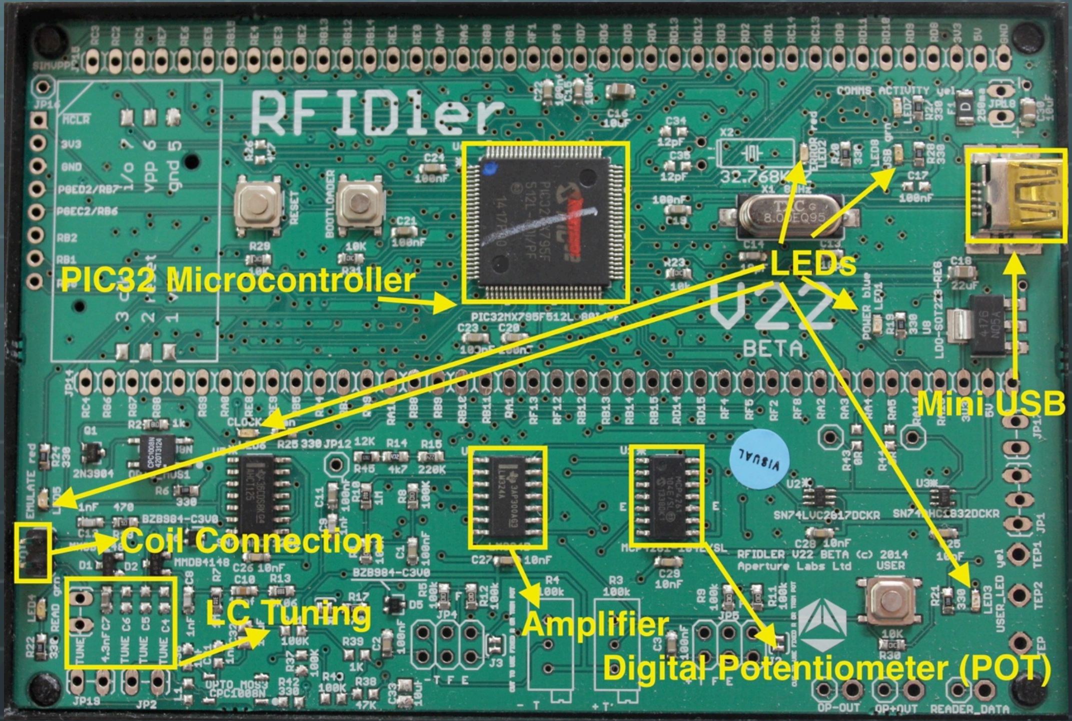
RFIDler

- 125kHz-134kHz ASK and FSK reading and emulation
- V22 Beta Obtained at DEF CON 22 (coil not pictured)



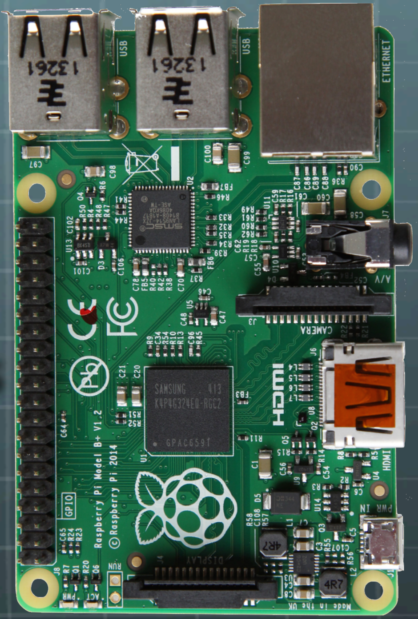
Annotated RFIDler

- Digital POTs for threshold tuning
- USB Serial Interface
- LED indicators
- PIC32 for main guts
- USB Powered



Raspberry Pi

- Popular embedded ARM dev kit
- Ready to go for Linux with available distros
- USB Support
  - Power
  - Data: 2 Ports, enough for RFID tool and WiFi dongle
- Close to 1” clearance in case

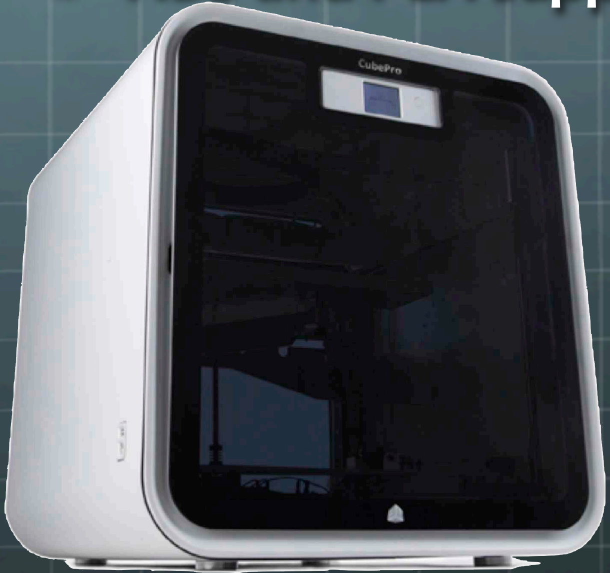


USB Armory

- USB based computer with security in mind
- ARM chipset ready to go for Linux
- Possibly smallest in class form factor
- USB
  - Power
  - Data

CubePro 3D Printer

- 70, 200, or 300 micron layer options
- ABS and PLA support with Nylon ‘coming soon’



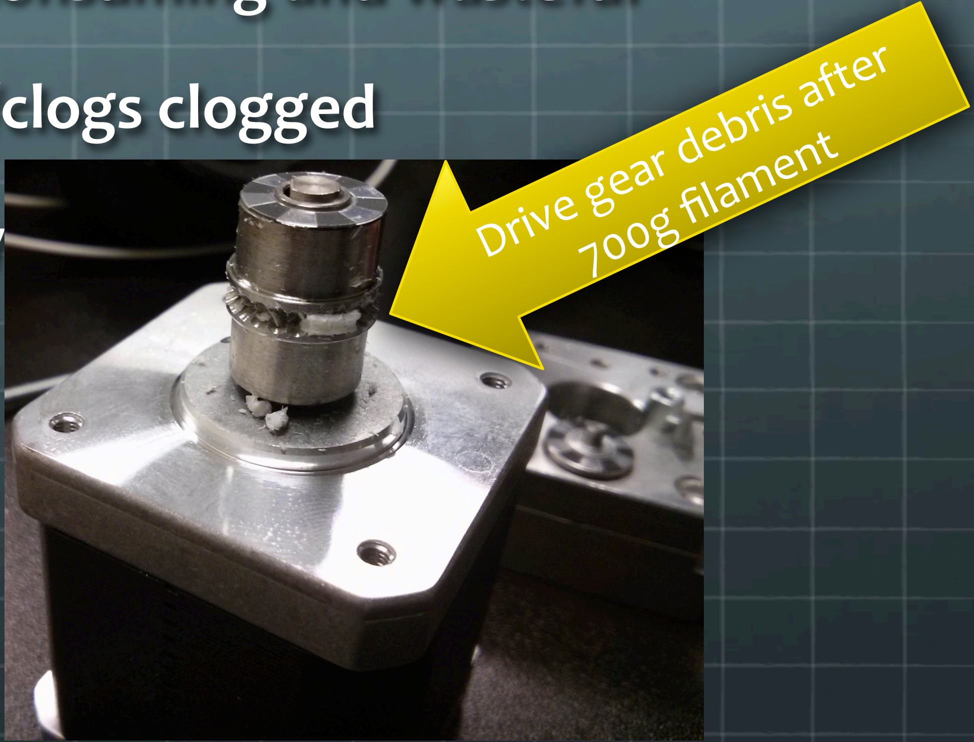
- 11.2” x 10.6” x 9.06” build volume
- Up to 3 print jets for complex builds

Cube Printing Process

- Drawings created using basic CAD tools
- CubePro software slices model into 2D layers
- Build plate receives a coating of water-soluble glue
- Gears feed filament into heated print jet
- Extruder moves on X and Y, plate moves on Z-axis
- Glue is dissolved in water and model scraped off

Printing Pitfalls

- Calibration can be time consuming and wasteful
- Extruders prone to jams/clogs clogged
- Industrial design is tricky
  - Physical limitations
  - Materials science



RFID Basics

- Low-Frequency: 125kHz – 134 kHz
  - Access Control Proximity Cards: HID, Indala, AWID, etc
  - Pet chips / implantable: AVID, Fecava, ISO 11784/11785
- High-Frequency: 13.56MHz
  - Access control: HID iCLASS, NXP Mifare family
  - NFC: Contactless payment, bluetooth pairing, etc.
  - IDs: ePA (German ID card), passports, etc

Passive RFID Tags

Tuned coil + IC = RFID Tag

- Coil tuned to carrier frequency (e.g. 125kHz)
- Reader energizes coil to power chip
- Chip modulates signal by adjusting damping
  - Damping controls how the circuit resonates

Common Modulations

FSK

- Frequency Shift Keying

PSK

- Phase Shift Keying

ASK

- Amplitude Shift Keying

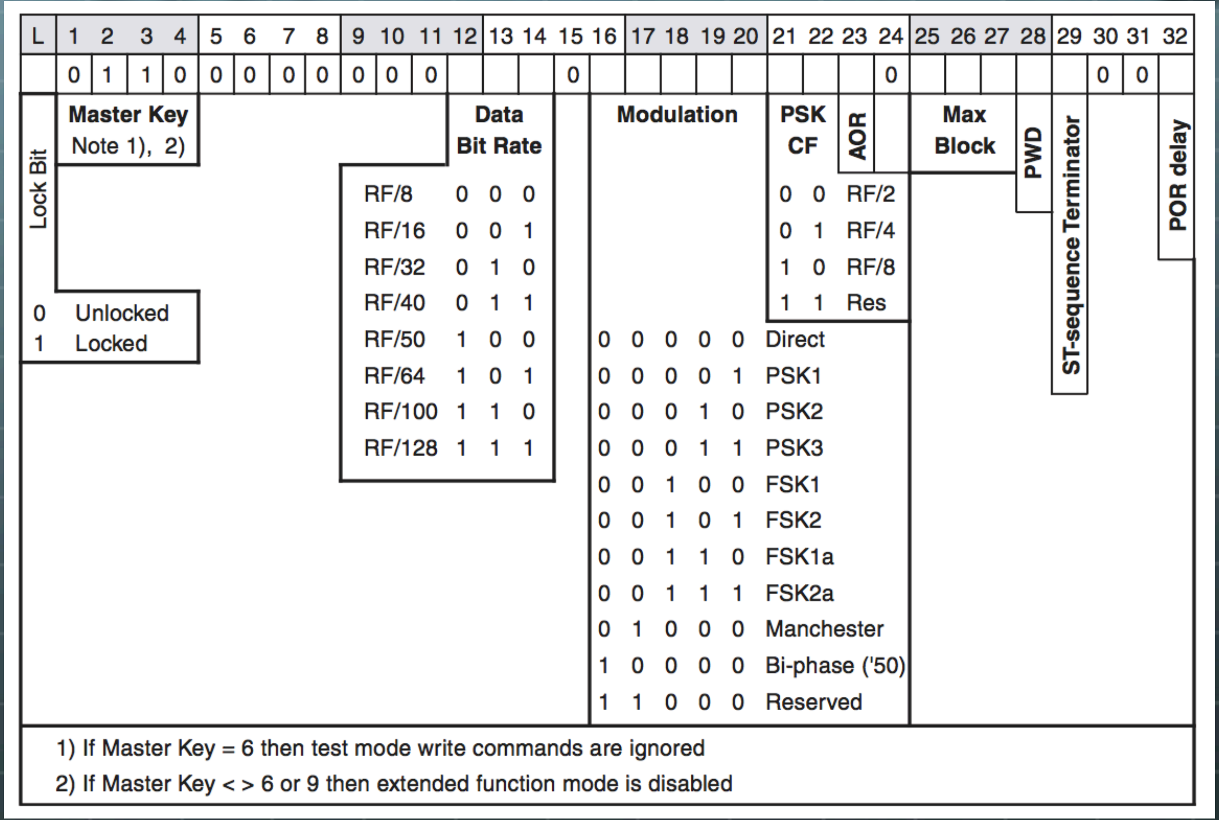
OOK

- On/Off Keying

T55xx Cards

- Low-Frequency tag with configurable properties
  - Modulation: FSK/PSK/ASK
  - Encoding: Manchester/Biphase
  - 8 4-byte storage blocks
    - 1 Config Block (0)
    - 7 Bitstream Blocks (1-7)

T5557 Block 0 Config



L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
	0	1	1	0	0	0	0	0	0	0	0				0									0						0	0	

Lock Bit

0	Unlocked
1	Locked

Master Key
Note 1), 2)

Data
Bit Rate

RF/8	0	0	0
RF/16	0	0	1
RF/32	0	1	0
RF/40	0	1	1
RF/50	1	0	0
RF/64	1	0	1
RF/100	1	1	0
RF/128	1	1	1

Modulation

0	0	0	0	0	Direct
0	0	0	0	1	PSK1
0	0	0	1	0	PSK2
0	0	0	1	1	PSK3
0	0	1	0	0	FSK1
0	0	1	0	1	FSK2
0	0	1	1	0	FSK1a
0	0	1	1	1	FSK2a
0	1	0	0	0	Manchester
1	0	0	0	0	Bi-phase ('50)
1	1	0	0	0	Reserved

PSK
CF

0	0	RF/2
0	1	RF/4
1	0	RF/8
1	1	Res

AOR

Max
Block

PWD

ST-sequence Terminator

POR delay

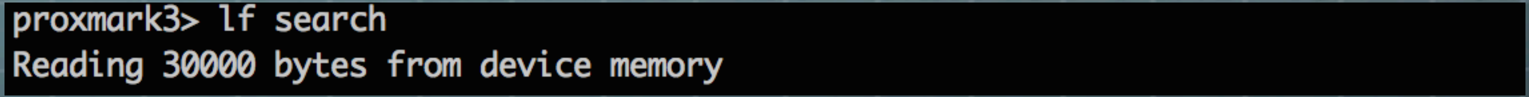
1) If Master Key = 6 then test mode write commands are ignored

2) If Master Key < > 6 or 9 then extended function mode is disabled

T5557 Block 0 (from datasheet)

T55xx Clone with PM3

Identify unknown tag

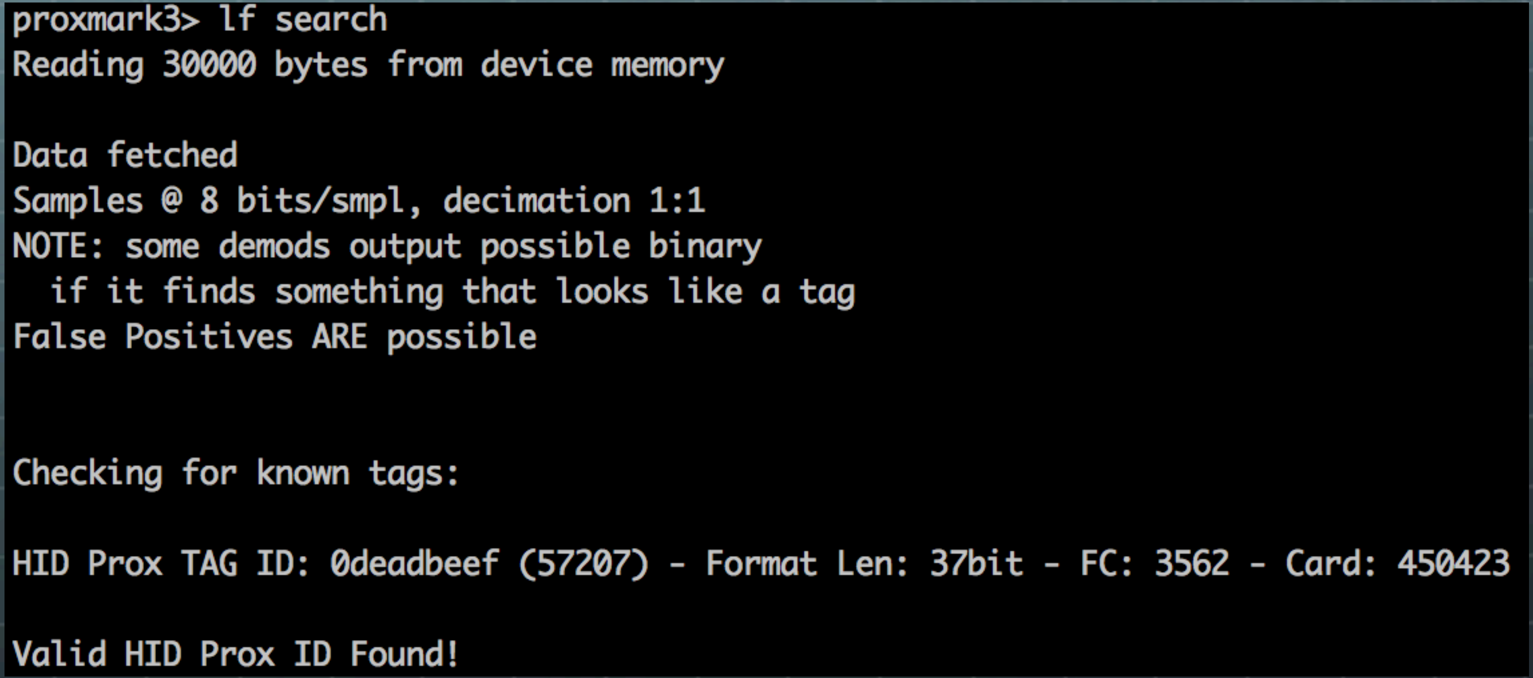


```
proxmark3> lf search
Reading 30000 bytes from device memory
```

- lf search : Automated Tag Detection by marshmellow
    - Step 1: lf read s
    - Step 2: data samples 30000
    - Step 3..n: Test each data *demod command until success

T55xx Clone with PM3

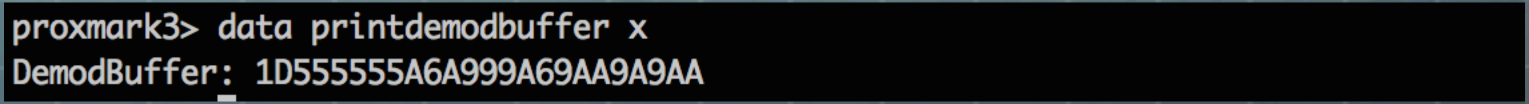
Identify unknown tag



```
proxmark3> lf search
Reading 30000 bytes from device memory

Data fetched
Samples @ 8 bits/smpl, decimation 1:1
NOTE: some demods output possible binary
  if it finds something that looks like a tag
False Positives ARE possible



Checking for known tags:

HID Prox TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423

Valid HID Prox ID Found!
```

T55xx Clone with PM3

Extract Tag Bitstream

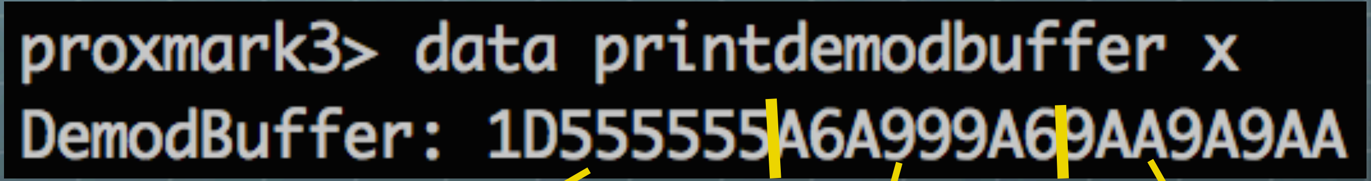


```
proxmark3> data printdemodbuffer x
DemodBuffer: 1D555555A6A999A69AA9A9AA
```

- DemodBuffer is the raw demodulated tag data
- Data can be split into 4-byte blocks for T55xx
- Bitstream starts with block 1

T55xx Clone with PM3

Print Tag's Demodulated Bitstream



```
proxmark3> data printdemodbuffer x
DemodBuffer: 1D555555A6A999A69AA9A9AA
```

Block 1
0x1D555555

Block 2
0xA6A999A6

Block 3
0x9AA9A9AA

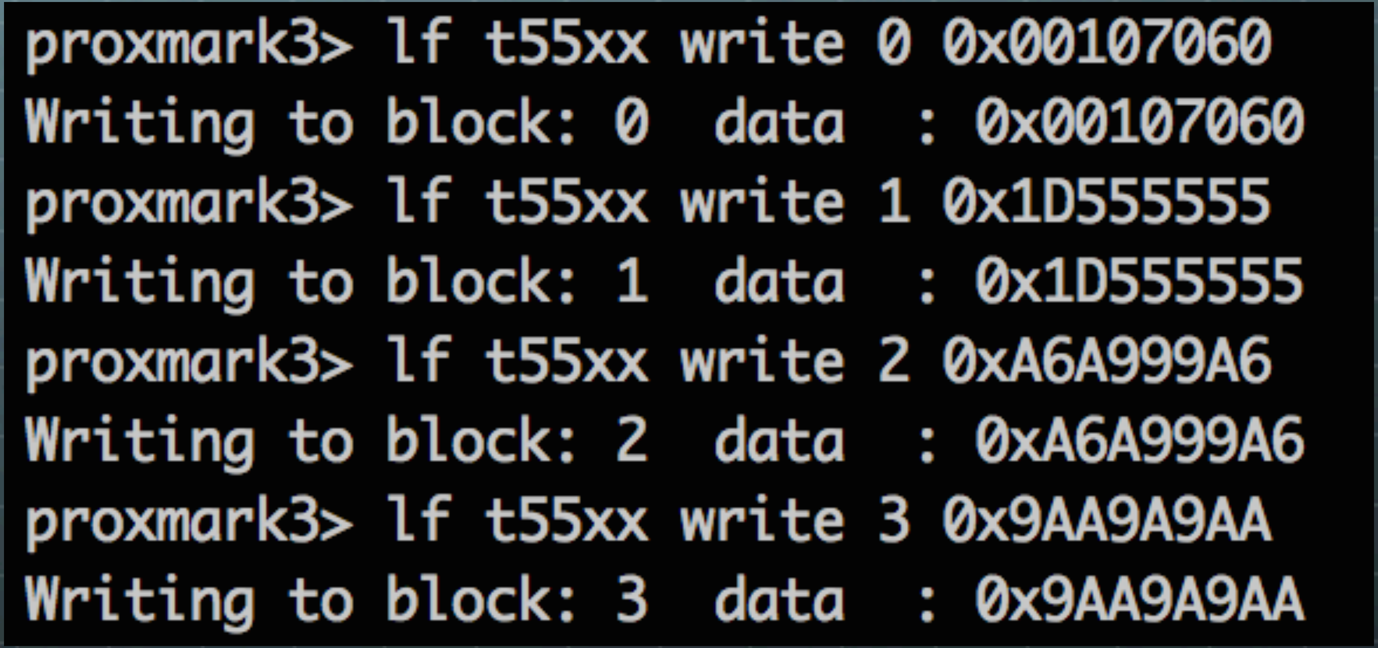
lf search & printdemodbuffer brought to you by Marshmellow

T55xx Block 0 Values

- Option 1: Determine block 0 manually via datasheets
- Option 2: Refer to excellent thread at proxmark.org
  - http://www.proxmark.org/forum/viewtopic.php?id=1767
- HID Tag uses: 0x00107060
  - 3 Blocks
  - FSK 2a Modulation
  - RF/50 Data Rate

T55xx Clone with PM3

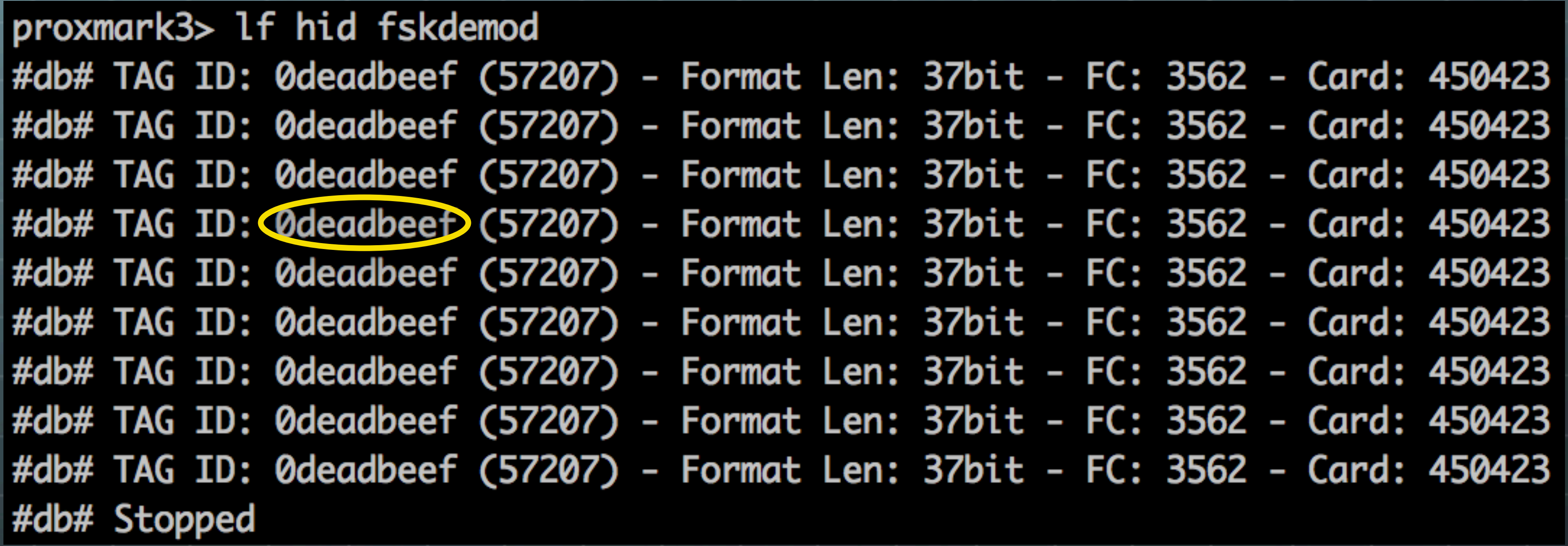
Programming a T5557 Card with HID 37 deadbeef



```
proxmark3> lf t55xx write 0 0x00107060
Writing to block: 0  data   : 0x00107060
proxmark3> lf t55xx write 1 0x1D555555
Writing to block: 1  data   : 0x1D555555
proxmark3> lf t55xx write 2 0xA6A999A6
Writing to block: 2  data   : 0xA6A999A6
proxmark3> lf t55xx write 3 0x9AA9A9AA
Writing to block: 3  data   : 0x9AA9A9AA
```

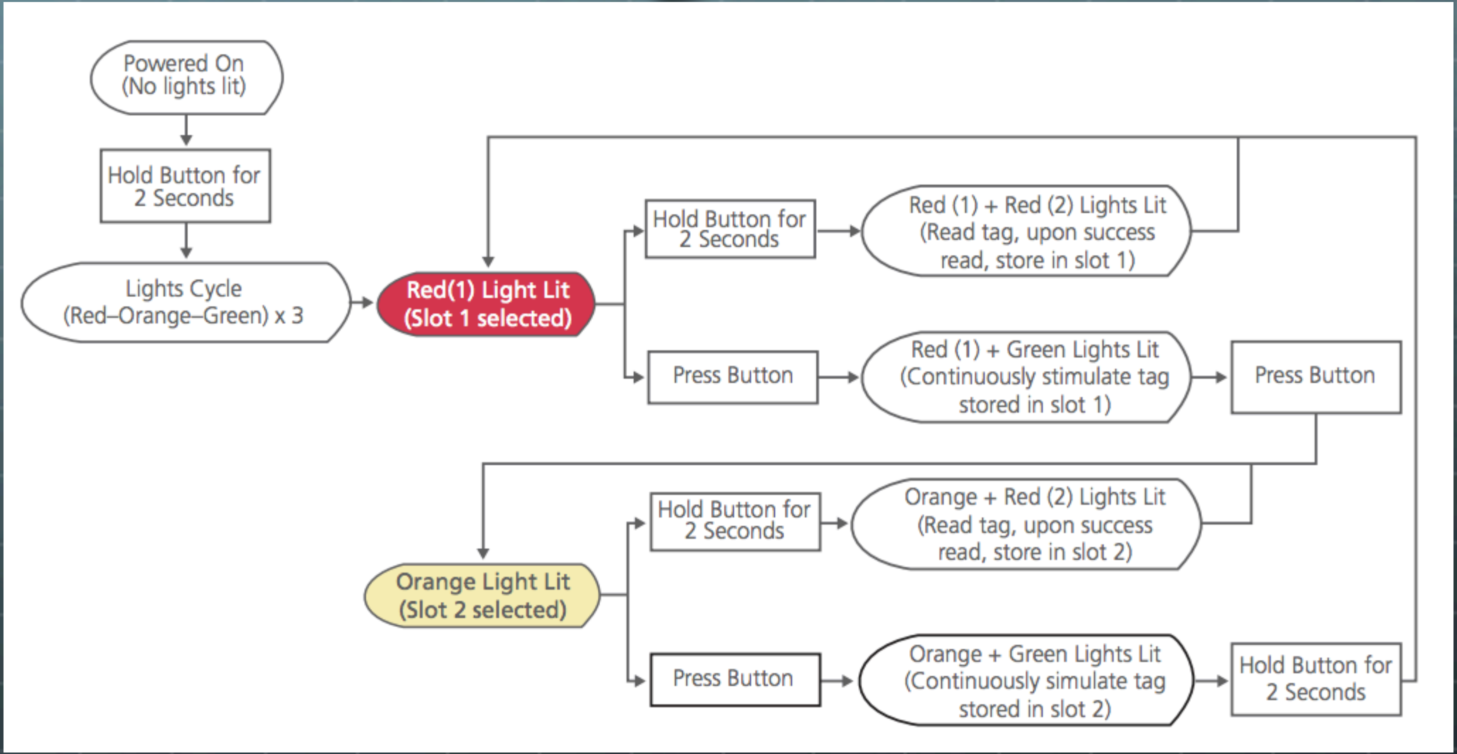
T55xx Clone with PM3

Trust But Verify (with lf hid fskdemod)



```
proxmark3> lf hid fskdemod
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# TAG ID: 0deadbeef (57207) - Format Len: 37bit - FC: 3562 - Card: 450423
#db# Stopped
```

Proxmark3 Stand-Alone


Powered On (No lights lit)
Hold Button for 2 Seconds
Lights Cycle (Red–Orange–Green) x 3
Red(1) Light Lit (Slot 1 selected)
Hold Button for 2 Seconds
Red (1) + Red (2) Lights Lit (Read tag, upon success read, store in slot 1)
Press Button
Red (1) + Green Lights Lit (Continuously stimulate tag stored in slot 1)
Press Button
Orange Light Lit (Slot 2 selected)
Hold Button for 2 Seconds
Orange + Red (2) Lights Lit (Read tag, upon success read, store in slot 2)
Press Button
Orange + Green Lights Lit (Continuously simulate tag stored in slot 2)
Hold Button for 2 Seconds

Default PM3 standalone operation flow from ProxBrute whitepaper

Stand-Alone Design

Using ARM CPU for independent operation

- **armsrc/appmain.c runs after init**
- **SamyRun() is invoked based on BUTTON_HELD(1000)**

CmdHIDdemodFSK and related functions used

- Capture/Replay/T55x7 Clone toggle with button
- Visual indication through LED changes

Offline HF Reading

iso14443a_setup(FPGA_HF_ISO14443A_READER_MOD)

iso14443a_select_card(uid, NULL, &cuid)

Return true means that a card select succeeded

Offline HF Simulation

UID is split into 2 4 byte values with valid byte order

SimulateIso14443aTag(1,uid_1st,uid_2nd,NULL)

UID only simulation (i.e. data support later?)

Offline HF Cloning

- Goal is to just set recorded UID on new tag
- MiFare Classic support only initially
  - hf mf csetuid <UID>
- Useful for systems relying on UID for auth
  - Samsung NFC locks, Smartphone unlock tags, etc.

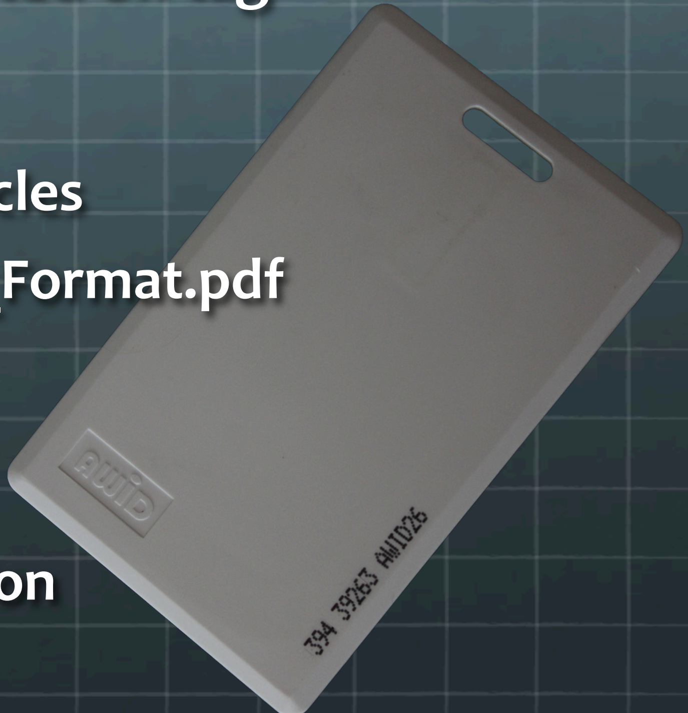
HF Stand-Alone Demo!

Adding PM3 Commands

- Target: ‘lf awid’ Proxmark3 context
  - fskdemod – Real-time demodulation for AWID tag
  - clone – Prepare T55xx card based on specified ID
  - sim – Tag emulation based on ID
- PM3 had no AWID26 code until this year
  - fskawiddemod added 2015
  - No lf awid context was added
  - Looked like a good place to contribute

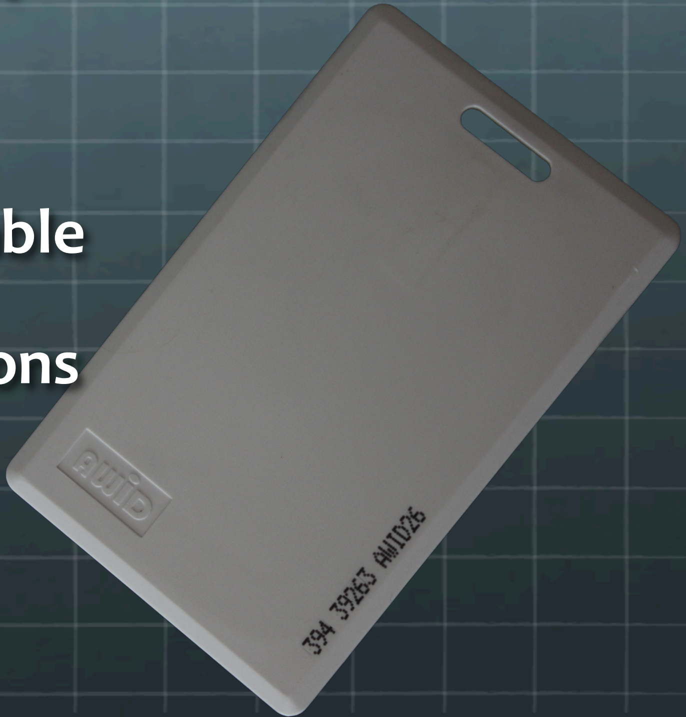
AWID26

- 8-bit Facility & 16-bit card number printed on tag
- 125kHz FSK2a (RF/50 data rate)
  - FSK Logic 0: 8 cycles, FSK Logic 1: 10 cycles
  - www.proxclone.com/pdfs/AWID_FSK_Format.pdf
- 96-bit transmission
  - Every 4th bit is a parity bit
  - 2 more parity bits for wiegand validation



Adding Commands

- Commands live in the proxmark3 client
- client/cmdlf.c for LF commands
- New commands added to CommandTable
- client/cmdlfawid.c created for definitions
  - Prepare & send UsbCommand to PM3



lf awid fskdemod

```
proxmark3> lf awid fskdemod
#db# AWID Found - BitLength: 26, FC: 1, Card: 31337 - Wiegand: 2f4d3, Raw: 011d8114eb2b28111111111
```

- Loop collecting samples and demodulating as AWID
- Runs as CmdAWIDdemodFSK() on ARM (armsrc/lfops.c)
- Structure copied from CmdHIDdemodFSK()
- CMD_AWID_DEMOD_FSK defined for USB command
- Logic in cmdlfawid.c exposes new command

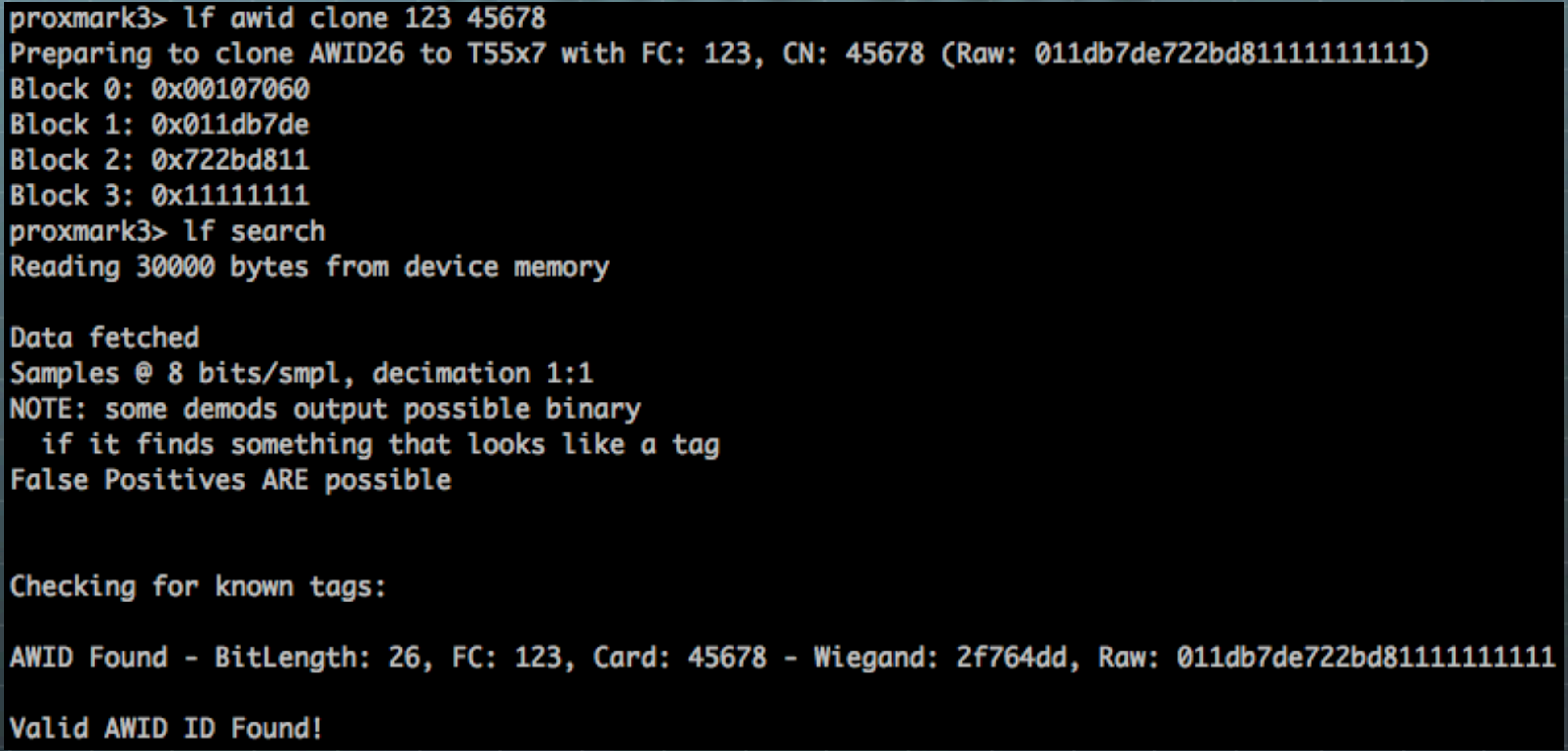
AWID26 BitStream

- Added getAWIDBits() utility function on client side
  - Translate facility-code and card number into bits
  - Card data stored as array of 12 uint8_t
  - Logic via AWID PDF
  - parityTest() via lfdemod.h
  - Hardcoded for 26-bit Wiegand

lf awid clone

- Programs T55x7 as specified AWID26
- Uses getAWIDBits() to determine BitStream
- Block 0 is static rate/modulation config
- Bitstream split into blocks 1-3
  - Cmd = CMD_T55XX_WRITE_BLOCK
  - Args: (data, block num, password)

lf awid clone

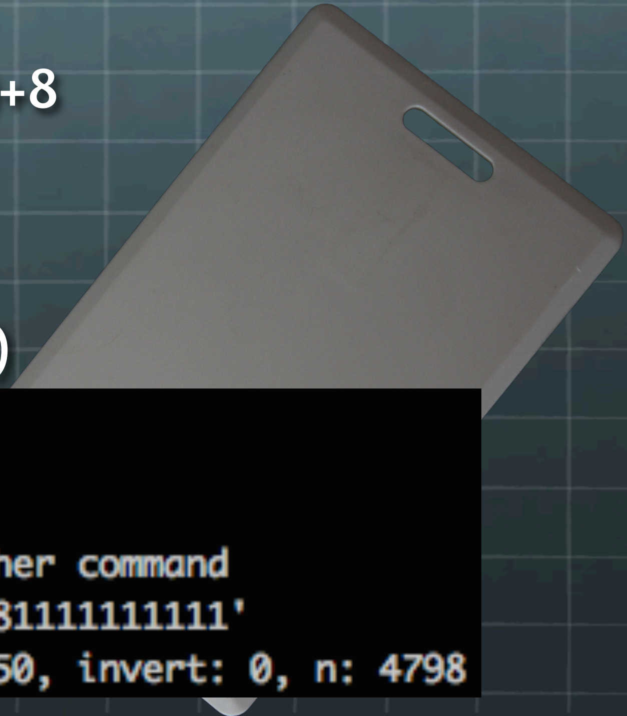


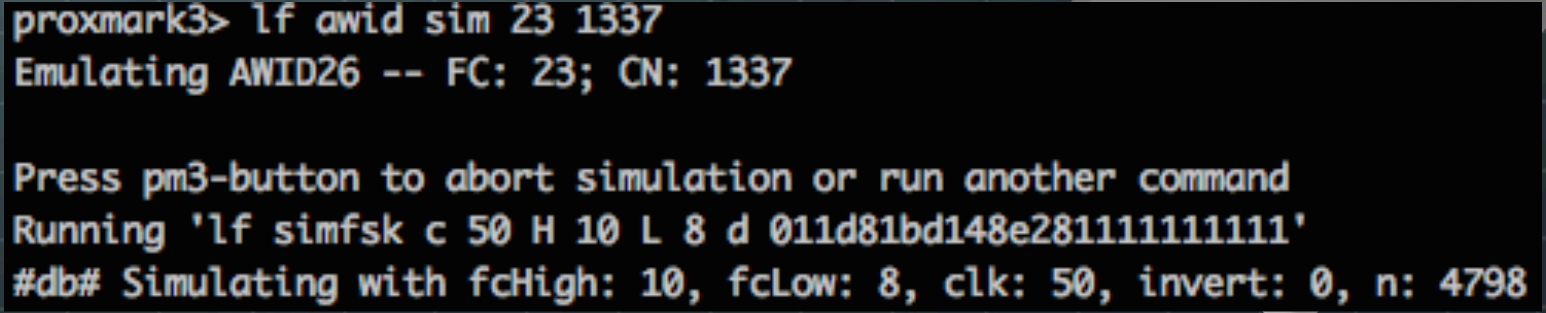
```
proxmark3> lf awid clone 123 45678
Preparing to clone AWID26 to T55x7 with FC: 123, CN: 45678 (Raw: 011db7de722bd8111111111)
Block 0: 0x00107060
Block 1: 0x011db7de
Block 2: 0x722bd811
Block 3: 0x11111111
proxmark3> lf search
Reading 30000 bytes from device memory

Data fetched
Samples @ 8 bits/smpl, decimation 1:1
NOTE: some demods output possible binary
  if it finds something that looks like a tag
False Positives ARE possible



Checking for known tags:

AWID Found - BitLength: 26, FC: 123, Card: 45678 - Wiegand: 2f764dd, Raw: 011db7de722bd8111111111

Valid AWID ID Found!
```

lf awid sim

Calculates/displays ‘lf simfsk’ parameters for AWID26

- cmd = CMD_FSK_SIM_TAG
- arg[0] = fcHigh<<8 + fcLow = (10<<8)+8
- arg[1] = CLK & Invert = 50
- arg[2] = Length = 96 (bits)
- d.asBytes = BitStream (1 bit per byte)





```
proxmark3> lf awid sim 23 1337
Emulating AWID26 -- FC: 23; CN: 1337

Press pm3-button to abort simulation or run another command
Running 'lf simfsk c 50 H 10 L 8 d 011d81bd148e281111111111'
#db# Simulating with fcHigh: 10, fcLow: 8, clk: 50, invert: 0, n: 4798
```

Antenna Construction

- The Problem
  - Traditional DIY involves a lot of trial and error
  - The outcome is not always ideal for practical attacks
- My Solution: 3D Printing
  - Create forms with precise measurements easily
  - Make coils that blend in

Supplies

- High AWG magnet wire (e.g. 40AWG)
- Utility knife (to remove enamel from wire ends)
- Soldering iron
- Heat shrink tubing (optional)
- Inductance meter (optional)

The Math of Tuning

- The antenna coil is an inductor (L)
- An LC circuit is formed by pairing it with a capacitor
- Goal is to match the coil’s L with the capacitor @ F

$$\boxed{F = \frac{1}{2\pi\sqrt{CL}}}$$

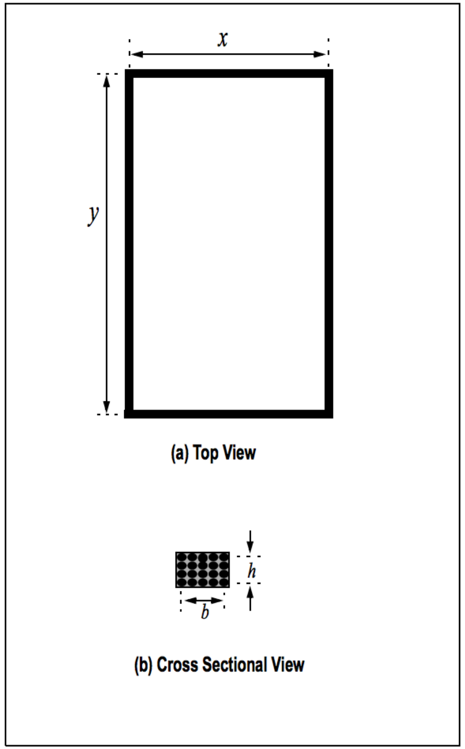
The Math of Tuning

- Microchip AN170
  - Coil induction equations
- Organized by coil shape
- For my 'multilayer rectangular loop coil':

$$L = \frac{0.0276(CN)^2}{1.908C + 9b + 10h}(\mu H)$$

WHERE→

FIGURE 11: N-TURN SQUARE LOOP COIL WITH MULTILAYER



Making a Fake Badge

- Started with D18c7db's LF design
  - Constructed with CD jewel cases cut to size
- Rectangular "badge-like" 70x40mm coil is perfect
  - Drew with CAD and exported for printing
  - Add an ID sticker for an extra convincing look
  - Lanyard clip allows route for discrete wiring

Winding the Coil

- Coil inductance is matched to capacitor for resonance
- First calculate L based on F and C

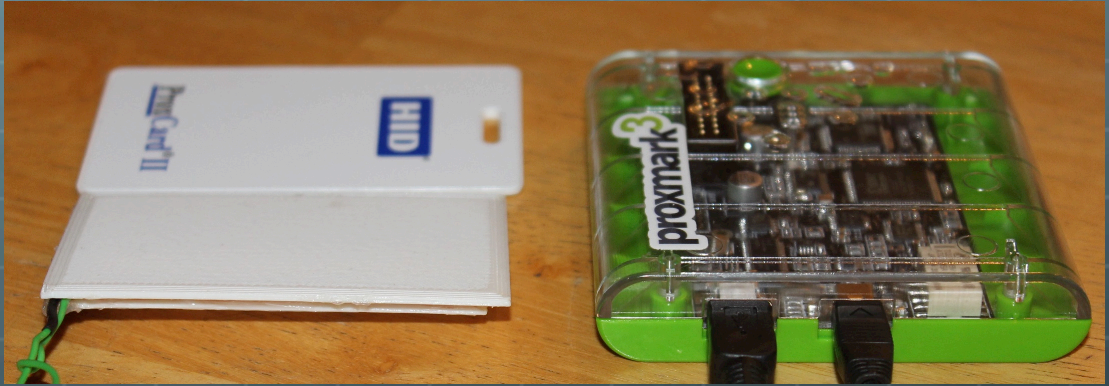
$$L = \frac{1}{4\pi^2 C F^2}$$

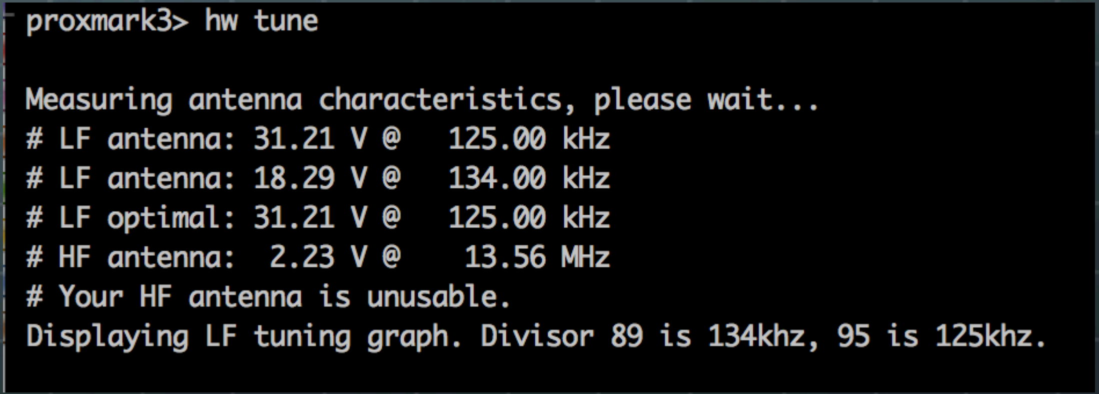
- Calculate turns required for L based on shape
  - For my Proxmark3: ~87 turns
  - For my RFIDler: ~57 turns
- Calculations only give guideline due to natural variance

Winding the Coil

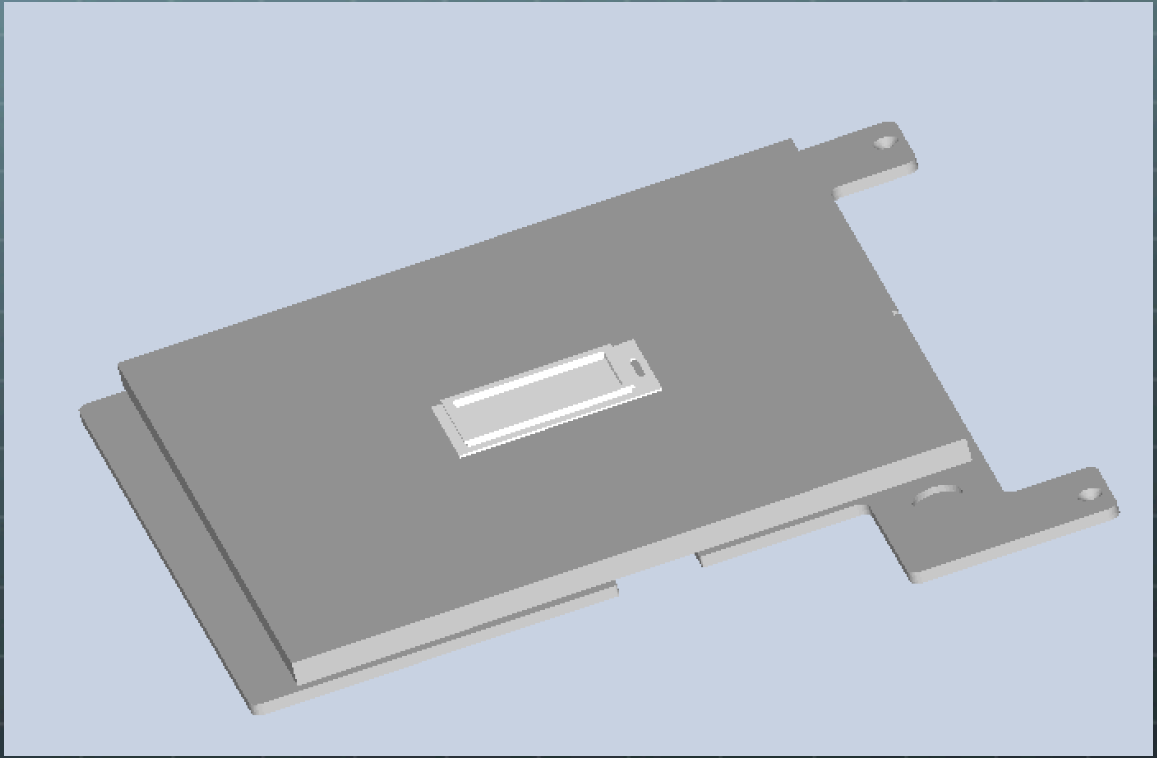
- Wind more turns than you expect to need
- Removing extra turns is easy
  - Making up for not enough turns – NOT SO EASY
- Count and watch for wire getting caught on edges

First Antenna Attempt




```
proxmark3> hw tune

Measuring antenna characteristics, please wait...
# LF antenna: 31.21 V @   125.00 kHz
# LF antenna: 18.29 V @   134.00 kHz
# LF optimal: 31.21 V @   125.00 kHz
# HF antenna:  2.23 V @    13.56 MHz
# Your HF antenna is unusable.
Displaying LF tuning graph. Divisor 89 is 134khz, 95 is 125khz.
```

Badge Revision



BADge-tenna

- $7.68 + $4.99 shipping for flexible white plastic
  - Suitable alternative for those without a 3d printer
- With CubePro: about $4 filament
- Cable hidden in black lanyard
- Scavanged RCA for length

Clipwned

Storage clipboards are great cover!

They blend well with room for toys!

Printed spacers hold the gear tight.

An official sticker seals the deal…

Clipboards lend authority, it’s up to you to take it!

What is Clipwned

Clipwned is a custom tool to grab RFID access control data quietly.

My Clippwned contains a proxmark connected to a USB armory for compact storage of all encountered tags.

The idea is to act official as if a survey taker – corporate stickers can help sell the story.

"Your name is spelled how?"
"Could I just grab that badge one second to make sure I've got your name right?"

Clipwned Supplies

- Amazon is full of “Storage Clipboards”
- Step 1 is finding one with the depth needed (3/4” minimum, 1” is great)
  - Check dimensions of tools going into clipwned
- 3D printed parts provide guides to keep components in place
  - Tape, wood, and other materials work as well

Making Clipwned

Attach battery to embedded computer for power

- Proxmark client compiles easily on many platforms
- libNFC supports major distros
- The RFIDler USB serial is almost universally supported

Connect embedded computer to RFID tool of choice via USB

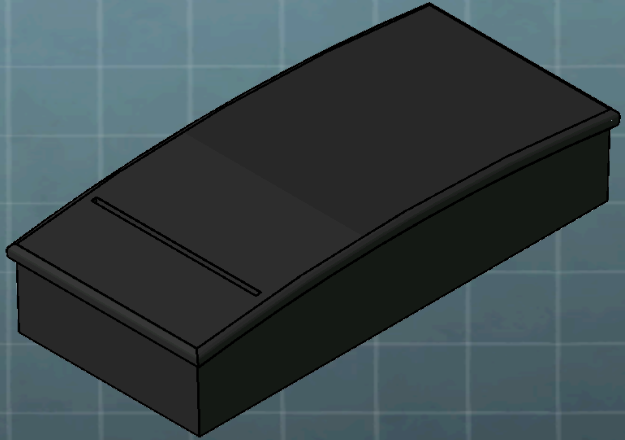
Cover stash with legitimate forms or blank pages

Make clipwned look like something a pro uses (notes, pens, etc.)

Making Fake Readers

- Building Information Modeling
  - Models available for common building components
  - This includes access control readers
  - Models are suited for modification and printing

Fake Readers



- Models via BIM make 3D printed readers possible
- Add an antenna inside for a hostile reader

- Place it over an existing reader and relay/record badge scans for a day

- Install rogue readers in unexpected places and see who scans
  - “Badge for the elevator? Cool upgrade!!!”

Phone Case Snooper

- Phone based NFC payment is being hailed for security
  - What is possible by having a coil hidden in a case?
    - Snooping on transactions for research?
    - Stealthy relay attacks?
    - Android Beam/NDEF attacks?
- Phone case models readily available on web
  - Cube even includes an iPhone case as a test file

Cloning MiFare

- MiFare is a family of tags not just a single tag type
  - MiFare Classic
  - MiFare Ultralight (and variations)
  - MiFare DESFire
  - MiFare DESFire EC2
  - Others?
- Tags support encryption in various capacities
  - All tags have a UID and sometimes that’s all that matters

“Magic” Cards

- Normally read-only UID is read-write instead
  - www.CloneMyKey.com (fast and reliable)
  - eBay “UID Changeable” (one vendor took 6+ weeks)
- Allows complete duplication of some card types
- This is why UID must not be used as a secure token

Ultralight Cloning

- NXP PN533 w/ libNFC and “Magic” MiFare Ultralight
- nfc-mfultralight r dump.mfd
    - Create MiFare dump of target card
- nfc-mfultralight w source.mfd
    - Write contents of source.mfd to changeable tag
- Ultralight doesn’t employ crypto

Working With RFIDler

- My RFIDler is not particularly reliable
- Printed case helps but card reads inconsistent
- POTSET commands have been most helpful
  - POTSET adjusts the digital threshold potentiometers
- Serial driven control best with embedded helper

Embedded Helpers

Pairing the RFIDler with a Raspberry Pi enables stealth & advanced attacks.

SD Storage

- Log stolen IDs

Networking

- Remote control

Automation

- Scriptable responses

Low-profile

- Small footprint

PM3 Stand-Alone Pt. 2

- Wouldn’t it be great it PM3 was wireless?
- Hooking up to an embedded PC does this for us.
- No need to expose pm3 if its hooked into WiFi
- Partner knows when read succeeded, sets modes, etc.
- All PM3 functionality is on the table without being seen!

Work In Progress (Thanks for Reading!)

Please join me Friday, Track 1 @ 18:00 for the rest!

Watch DEFCON and Triwpire sites for full slides & whitepaper…

Craig Young
Security Researcher, Tripwire VERT
Twitter: @CraigTweets