给平民的MouseJack – Qcon 2016

Presenter: kj

Language Selection

Introduction

Who am I

Almost Every Weekend

With VN Security since year 2009

- CTF player
- Weekend gamer

Most of the time

Running xandora.net project.

- I am the coder
- I am the administrator
- I am what I am

Once a year

Hack in the box die hard fans

- Good friends
- CTF CTF and CTF

- 2008, Hack In The Box CTF Winner
- 2010, Hack In The Box Speaker, Malaysia
- 2012, Codegate Speaker, Korea
- 2015, VXRL Speaker, Hong Kong
- 2015, HITCON CTF, Prequal Top 10
- 2016, Codegate CTF, Prequal Top 5

- OSX, Local Privilege Escalation
- Code commit for metasploit 3
- GDB Bug hunting
- Linux Randomization Bypass
- http://www.githiub.com/xwings/tuya

Research Credits

Bastille

REEBUF

黑客与极客

WooYun SECURITY

0xBC

NORDIC SEMICONDUCTOR

Smarter Things

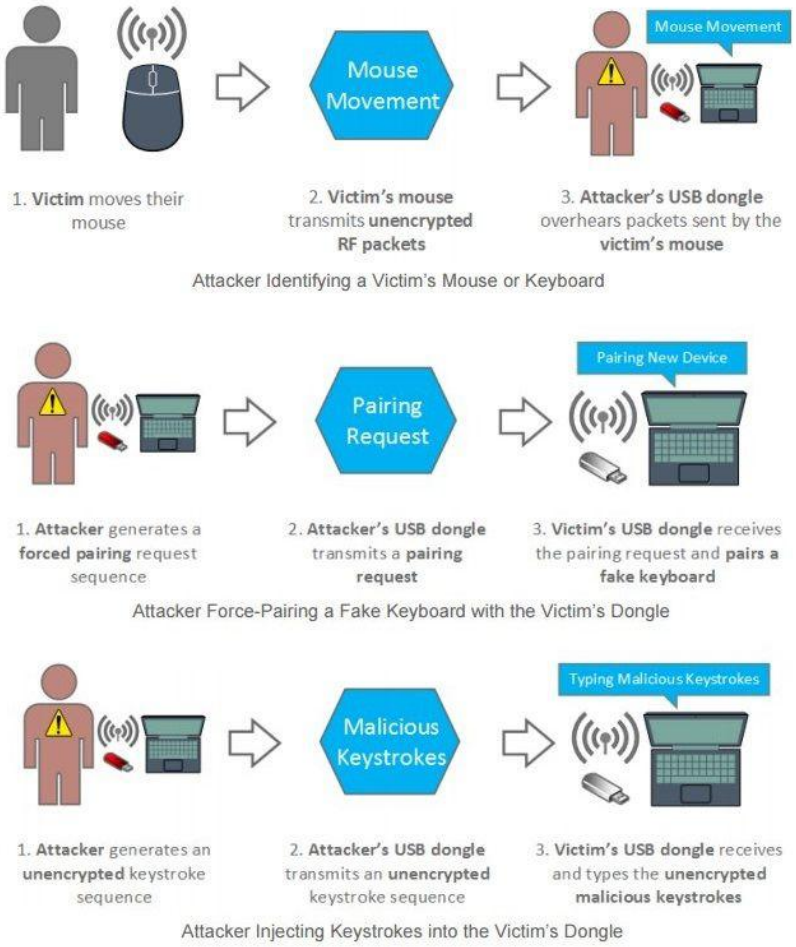
hackaday.io

Dangerous Prototypes

- Partner In Crime: klks84, https://twitter.com/klks84
- All my missing Logitech keyboard and mouse

What Is MouseJack

What Is MouseJack



- Targeting non-Bluetooth keyboard and mice
- Sniff and transmit special crated radio packet towards victims
- Keyboards normally sends encrypted packets
- Affected Product ? Most of the non-Bluetooth keyboard and mouse

How It Works

1. TERM=screen-256color-bce tmux (tmux)

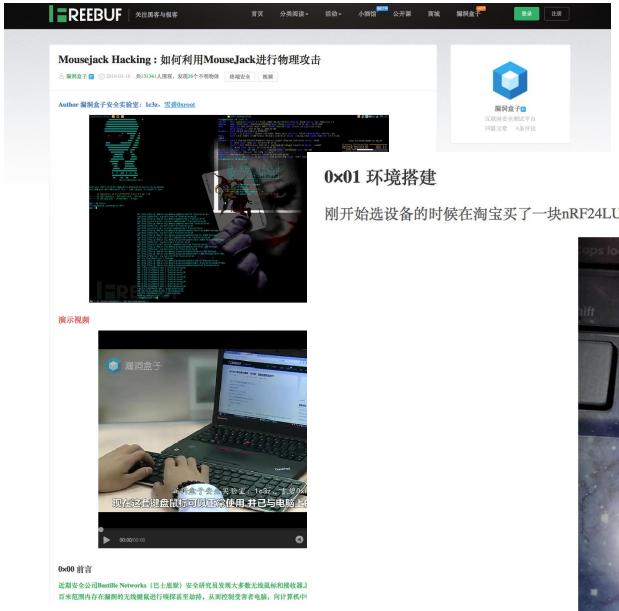
```
[2016-02-25 12:53:33.042]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.058]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.065]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.066]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.074]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.082]  17  22  1F:C9:91:16:07  00:D3:73:9A:AA:B9:F8:9F:BB:66:A6:59:11:FF:00:00:00:00:00:00:00:F6
[2016-02-25 12:53:33.083]  17  22  1F:C9:91:16:07  00:40:00:08:B8:B9:4A:EC:1A:67:A6:59:11:FE:00:00:00:00:00:00:00:E1
[2016-02-25 12:53:33.126]  17   5  1F:C9:91:16:07  00:40:01:18:A7
[2016-02-25 12:53:33.126]  17   5  1F:C9:91:16:07  00:D3:73:9A:AA
[2016-02-25 12:53:33.198]  17  22  1F:C9:91:16:07  00:D3:41:6B:76:DB:87:BF:C4:C1:A6:59:12:00:00:00:00:00:00:00:00:54
[2016-02-25 12:53:33.206]  17  22  1F:C9:91:16:07  00:D3:11:E6:B0:5F:AF:05:55:43:A6:59:12:01:00:00:00:00:00:00:00:C9
[2016-02-25 12:53:33.207]  17  22  1F:C9:91:16:07  00:D3:73:9A:AA:B9:F8:9F:BB:66:A6:59:11:FF:00:00:00:00:00:00:00:F6
[2016-02-25 12:53:33.221]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.221]  17   5  1F:C9:91:16:07  00:D3:11:E6:B0
[2016-02-25 12:53:33.237]  17   5  1F:C9:91:16:07  00:D3:73:9A:AA
[2016-02-25 12:53:33.245]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.246]  17   5  1F:C9:91:16:07  00:D3:11:E6:B0
[2016-02-25 12:53:33.262]  17  22  1F:C9:91:16:07  00:D3:E4:C0:A7:12:04:2D:A4:75:A6:59:12:03:00:00:00:00:00:00:00:72
[2016-02-25 12:53:33.263]  17  22  1F:C9:91:16:07  00:40:00:08:B8:DB:87:BF:C4:C1:A6:59:12:00:00:00:00:00:00:00:00:54
[2016-02-25 12:53:33.278]  17  22  1F:C9:91:16:07  00:D3:E8:52:91:51:2B:01:35:71:A6:59:12:05:00:00:00:00:00:00:00:29
[2016-02-25 12:53:33.286]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.286]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.374]  17   5  1F:C9:91:16:07  00:40:01:18:A7
[2016-02-25 12:53:33.386]  17  22  1F:C9:91:16:07  00:D3:11:B4:68:5D:ED:20:2B:B8:A6:59:12:06:00:00:00:00:00:00:00:9C
[2016-02-25 12:53:33.386]  17  22  1F:C9:91:16:07  00:40:00:08:B8:DB:87:BF:C4:C1:A6:59:12:00:00:00:00:00:00:00:00:54
[2016-02-25 12:53:33.402]  17  22  1F:C9:91:16:07  00:D3:09:D0:54:2A:B0:EE:15:E8:A6:59:12:08:00:00:00:00:00:00:00:22
[2016-02-25 12:53:33.403]  17  22  1F:C9:91:16:07  00:D3:11:B4:68:5D:ED:20:2B:B8:A6:59:12:06:00:00:00:00:00:00:00:9C
[2016-02-25 12:53:33.409]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.425]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.425]  17   5  1F:C9:91:16:07  00:D3:11:B4:68
[2016-02-25 12:53:33.441]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.442]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.465]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.466]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.482]  17  10  1F:C9:91:16:07  00:4F:00:01:18:00:00:00:00:98
[2016-02-25 12:53:33.482]  17  10  1F:C9:91:16:07  00:40:00:08:B8:5D:ED:20:2B:B8
[2016-02-25 12:53:33.597]  17  22  1F:C9:91:16:07  00:D3:9C:95:87:48:F2:8C:04:3F:A6:59:12:0C:00:00:00:00:00:00:00:4F
[2016-02-25 12:53:33.604]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.604]  17   5  1F:C9:91:16:07  00:40:00:08:B8
[2016-02-25 12:53:33.621]  17  22  1F:C9:91:16:07  00:D3:52:98:C8:56:A6:06:4D:55:A6:59:12:0D:00:00:00:00:00:00:00:B9
[2016-02-25 12:53:33.622]  17  22  1F:C9:91:16:07  00:40:00:08:B8:00:00:00:00:98:AE:59:12:0B:00:00:00:00:00:02:00:8A
```

1 gyaresu 0 python 1 jupyter 2 ..tual IRC Logs 12:53 25 Feb zaphod

- Scan all the nearby wireless mouse, signal jumping
- Sniff targeted victim
- Dump “keystroke”, ”Mouse Stroke”
- Replay, Hijack, 0wn3d

Objectives

Why This Research



- Most complete MouseJack implementation guide, in chinese
- Both guide based on Crazyradio. “PA” and non “PA”

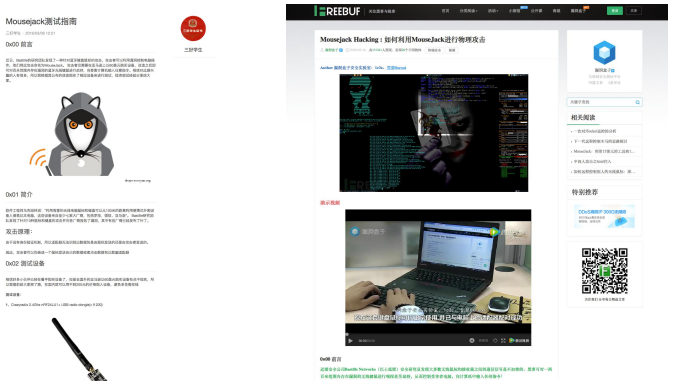
- Objective 1: Can it be cheaper?
- Objective 2: Smaller? (Not too obvious)
- Objective 3: Easier to purchase? Just tabao it?

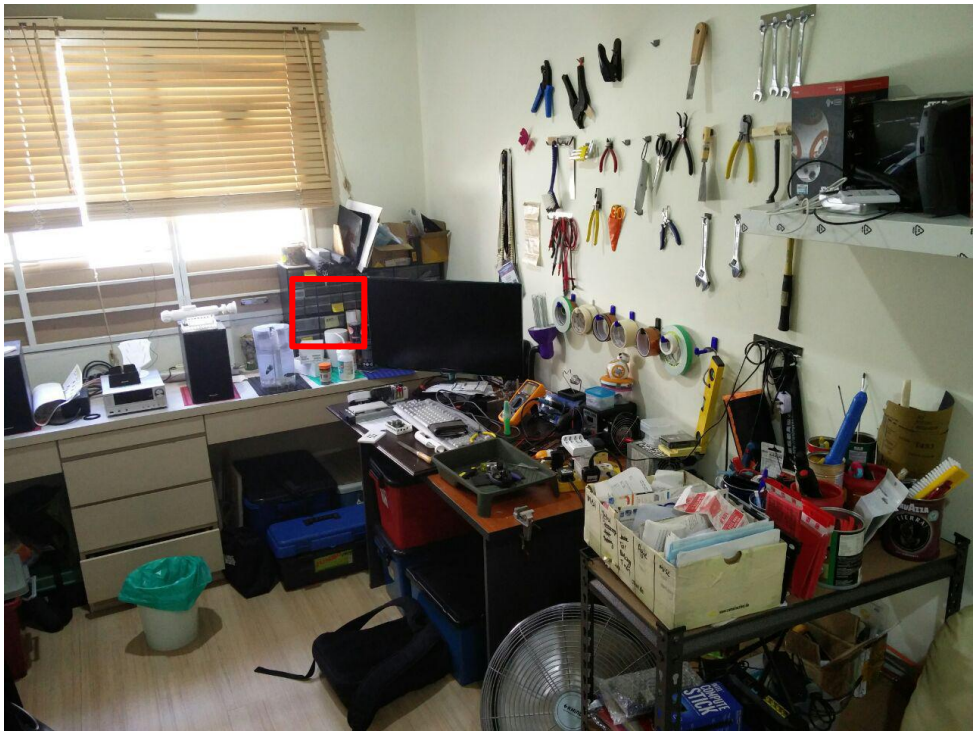
What Not, But

- Nothing to do with keyboard injection
- Nothing to do with breaking keyboard encryption
- Nothing to do with mouse injection
- Nothing to do with super long distance sniffing
- But, We do have something. Yes, that something

How It Started

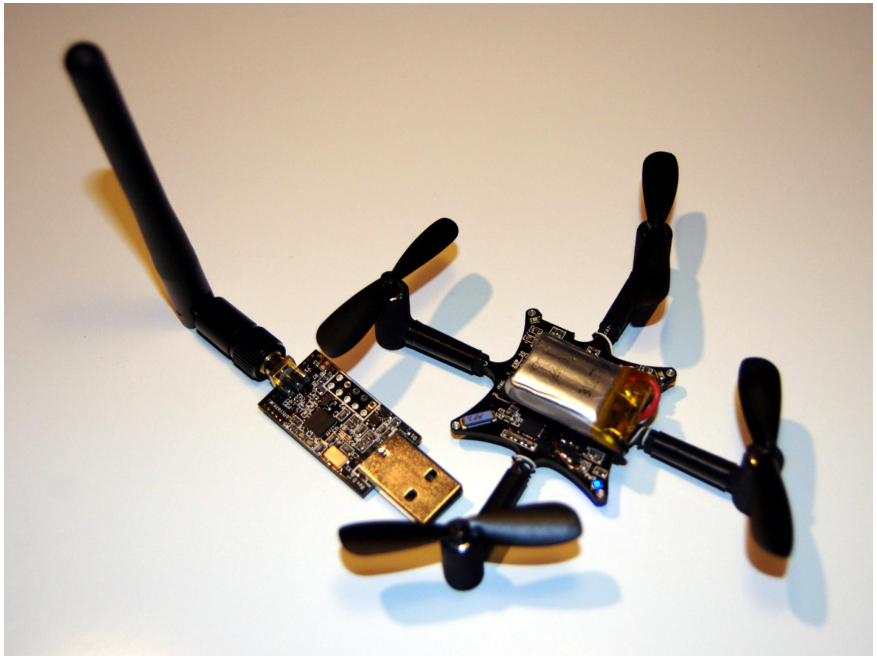
It Does Ring the Bell





- Got it few years back
- Always hide in small little corner
- After two times flying
- I think I give up, completely

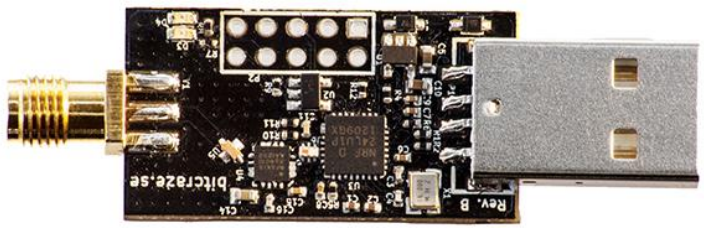
The Crazyradio

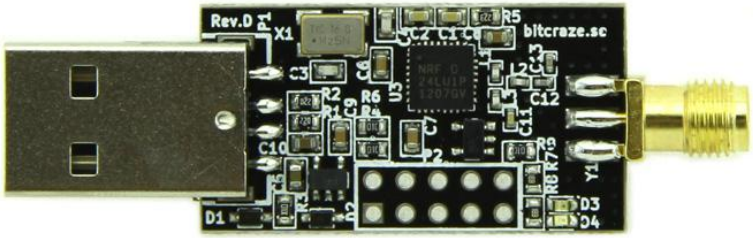


- Based on nRF24LU1+ chip
- 125 radio channels
- Send and receive packets up to 32 bytes
- Design by bitcraze.io to fly crazyflies
- Comment: Not too easy to fly

In The Beginning, I Screw It Up

Two Different Types, PA and Non PA





- In the beginning, there are two types of crazyradio
- Item 1. Crazyradio PA
- Item 2, Crazyradio (Obsolete)
- Crazyradio PA comes with extended range. 1KM
- Bitcraze no longer selling crazyradio, only PA model is available

How It All Started

bitcraze / crazyradio-firmware

Watch 22 | Star 41 | Fork 36

Code | Issues 10 | Pull requests 0 | Wiki | Pulse | Graphs

Releases | Tags

Latest release

0.53

b197536

0.53

ataffanel released this on Nov 17, 2014 · **22 commits** to master since this release

Added Crazyradio PA support with a compile flag.

Flash cradio-0.53.bin on Crazyradio and cradio-pa-0.53.bin on Crazyradio PA.

No added functionality for Crazyradio.

Downloads

- cradio-0.53.bin 5.66 KB
- cradio-pa-0.53.bin 5.67 KB
- Source code (zip)
- Source code (tar.gz)

on Jun 14, 2013 — **0.52** 98574e7 zip tar.gz

Since on May 8, 2013 — Show 2 other tags

on Feb 3, 2013 — **v0.4** fa33302 zip tar.gz

USB bootloader (command line instructions)

Please note that you might have to exchange *python* with *python2* if you distro uses python3.

First Crazyradio has to be rebooted in USB bootloader mode. To do so insert the dongle in the pc, open a terminal window and run the bootloader launcher:

```
> cd crazyradio-firmware
> python usbtools/launchBootloader.py
Launch bootloader .
Bootloader started
```

After running this tool the Crazyradio dongle should have disappeared and a new device named **nRF24LU1P-F32 BOOT LDR** should appear.

To flash the firmware use the nrfbootload.py script:

```
> cd crazyradio-firmware
> python usbtools/nrfbootload.py flash cradio-0.53.bin
Found nRF24LU1 bootloader version 18.0
Flashing:
  Flashing 5810 bytes...
Flashing done!
Verifying:
  Reading cradio-pa-0.53.bin...
  Reading 5810 bytes from the flash...
Verification succeded!
```

- At the beginning, there are two crazyradio
- Flashing the “PA” firmware in to the the NON “PA” is a bad idea
- Somehow, boot loader been overwritten
- The End

Solutions

Using BusPirate

It's possible to re-program the Crazyradio using a BusPirate and this script via SPI.

Couple of caveats:

- Tested only on OS X. Should work on Linux without modification, and Windows with very minor changes to use the windows serial module.
- It's very slow (~5 minutes to flash the entire .bin file). I deemed this acceptable as this script is for emergency recovery only. I can make it faster if necessary.

Prerequisites:

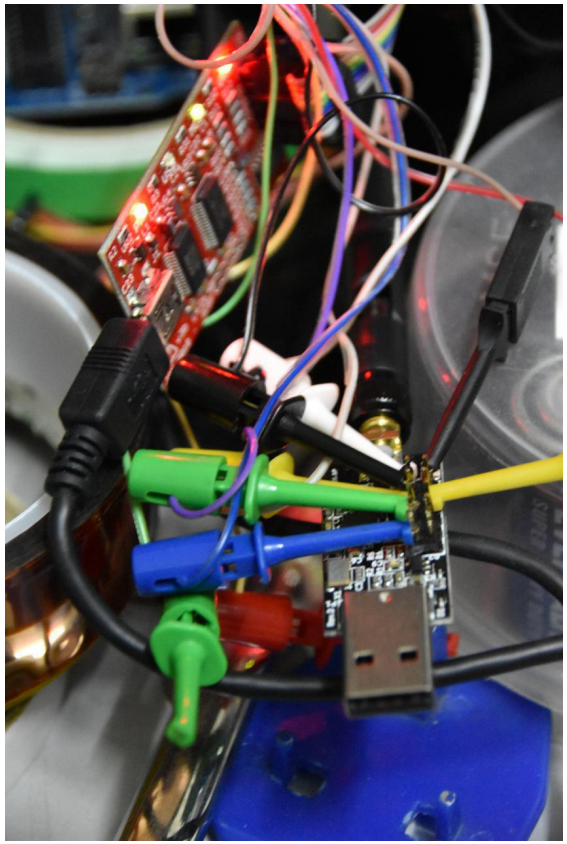
- A Bus Pirate (you should know where to get one of these, if you don't already have one).
- perl and either Device::SerialPort (*nix) or Win32::SerialPort (Windows)
- Some jumper wires to connect the SPI lines on the radio to the ones on the Bus Pirate.

Instructions:

1. Solder a 2×5 pin header onto the programming port of the crazyradio. There's an unpopulated footprint already there for you.
2. Connect the crazyradio to your Bus Pirate using the table below (also noted in the script and readme on git)

```
Bus Pirate       CrazyRadio
==========================
MOSI ()      ->  MOSI  (6)
MISO ()      ->  MISO  (8)
SCK  ()      ->  SCK   (4)
CS   ()      ->  CS    (10)
AUX  ()      ->  PROG  (2)
3V3  ()      ->  3V3   (5)
GND  ()      ->  GND   (9)
```

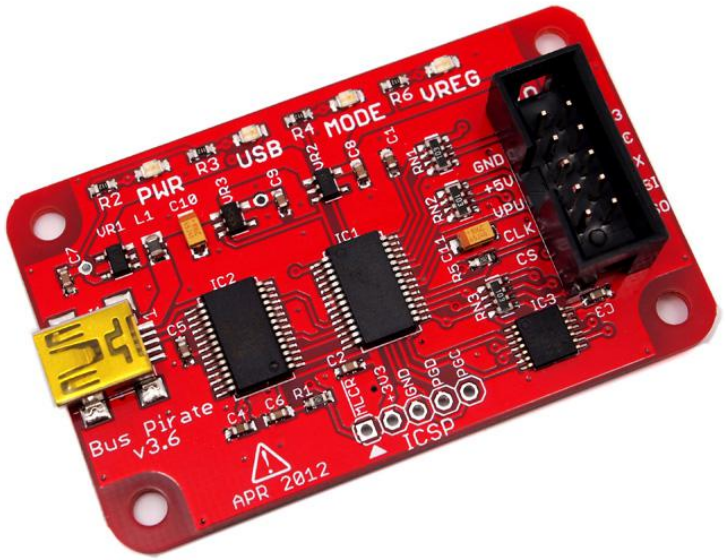
3. Run the script: perl ./flasher.pl -input ./cradio-0.51.bin -device [serial device]
4. Wait till you see lots of hex addresses crawling up your screen. Your device is programming.
5. Go make a sandwich or have a beer (or both).

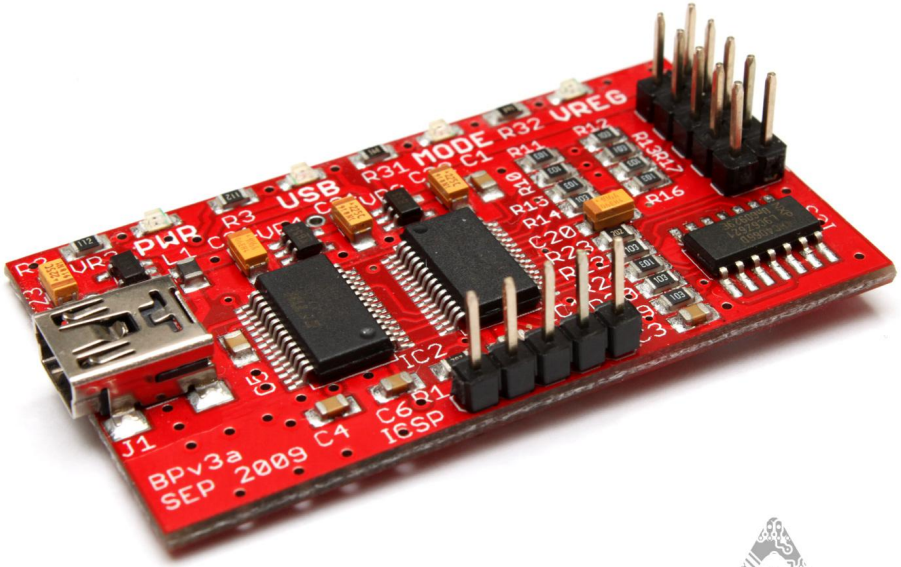


- Crazyradio wiki says need a complete SPI flash
- 1 Unit Bus Pirate
- 1 Unit 2 x 5 Pins
- Almost stable hand

Tools Needed

What Is Bus Pirate





Dangerous Prototypes

- Item 1, version 3.6
- Item 2, version 4
- Support 1-Wire, I2C, SPI, JTAG, Asynchronous Serial, MIDI and etc
- SPI is what we need
- Sells by seeeds studio and not in taobao

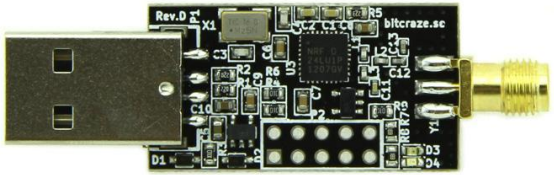
The Perl Script

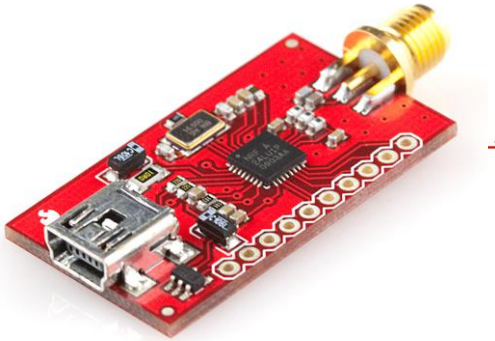
```
#!/usr/bin/perl -w

# Simple perl script to drive the Bus Pirate and unbrick your CrazyRadio dongle.
# Adapted (sorta) from the Bus Pirate example script and mbed NRF24LU1+ flasher projects:
#  http://code.google.com/p/the-bus-pirate/source/browse/trunk/scripts/SPIeeprom.pl
#  http://mbed.org/users/mux/code/nrflash
#
# This script uses the aux output on the Bus Pirate as the PROG pin on the CrazyRadio's NRF24LU1+ chip.
#
# Electrical connections are as follows:
#
#  Bus Pirate       CrazyRadio
#  ==========================
#  MOSI ()      ->  MOSI  (6)
#  MISO ()      ->  MISO  (8)
#  SCK  ()      ->  SCK   (4)
#  CS   ()      ->  CS    (10)
#  AUX  ()      ->  PROG  (2)
#  3V3  ()      ->  3V3   (5)
#  GND  ()      ->  GND   (9)

use strict;
use feature 'say';
use Getopt::Long;
use Device::SerialPort;
use Time::HiRes qw/usleep/;

use constant {
    WREN        => "\x06",
    WRDIS       => "\x04",
    RDSR        => "\x05",
    WRSR        => "\x01",
    READ        => "\x03",
    PROGRAM     => "\x02",
    ERASE_PAGE  => "\x52",
    ERASE_ALL   => "\x62",
    RDFPCR      => "\x89",
    RDISMB      => "\x85",
    ENDEBUG     => "\x86",
    RDYN        => "\x10",
    FLASH_LEN   => 32768,

    BP_CS       => "\x01",
    BP_AUX      => "\x02",
    BP_PULLUP   => "\x04",
    BP_POWER    => "\x08",
};

my %opts;
my $port;
my $time = 500;
my $status_byte;
my $return;
```

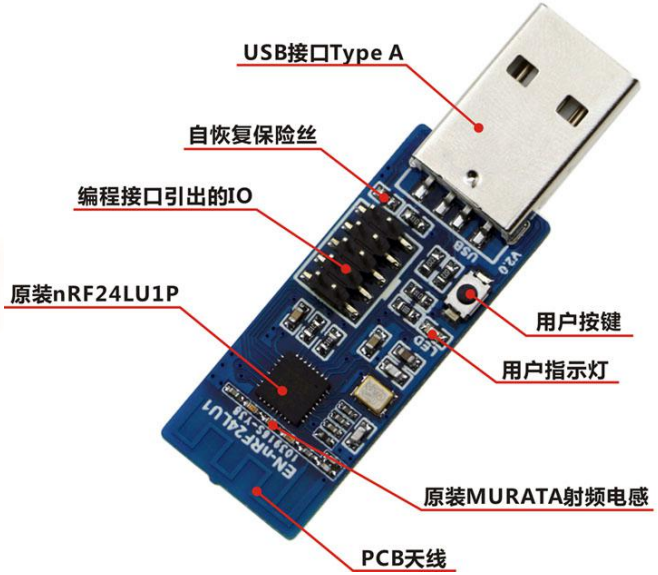
- https://github.com/xwings/tuya
- The defector standard SPI flashing script for crazyradio

Before We Go Further

nRF24LU1+ Based Device

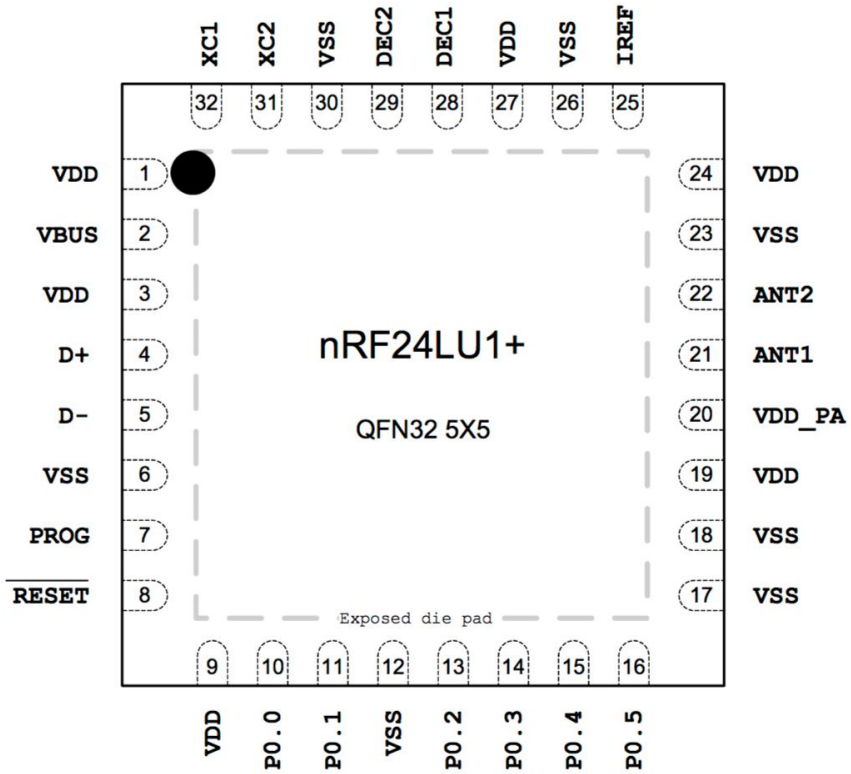






- Comes with nRF24L01 + 2.4 GHz RF transceiver
- USB Connector
- External Antenna or PCB Antenna

nRF24LU1+ Specification



NORDIC SEMICONDUCTOR
Smarter Things

- nRF24LU1 + 2.4 GHz RF transceiver
- Full speed USB 2.0 compliant device controller
- 8-bit microcontroller
- 16 or 32 kilobytes of flash memory
- Up to 12 Mbps air data rate
- Comes with AES encryption acceleration
- Full Spec document in: https://github.com/xwings/tuya

Saving Crazyradio

Soldering

Using BusPirate

It's possible to re-program the Crazyradio using a BusPirate and this script via SPI.

Couple of caveats:

- Tested only on OS X. Should work on Linux without modification, and Windows with very minor changes to use the windows serial module.
- It's very slow (~5 minutes to flash the entire .bin file). I deemed this acceptable as this script is for emergency recovery only. I can make it faster if necessary.

Prerequisites:

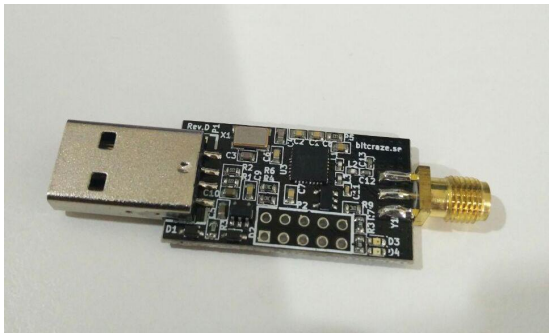
- A Bus Pirate (you should know where to get one of these, if you don't already have one).
- perl and either Device::SerialPort (*nix) or Win32::SerialPort (Windows)
- Some jumper wires to connect the SPI lines on the radio to the ones on the Bus Pirate.

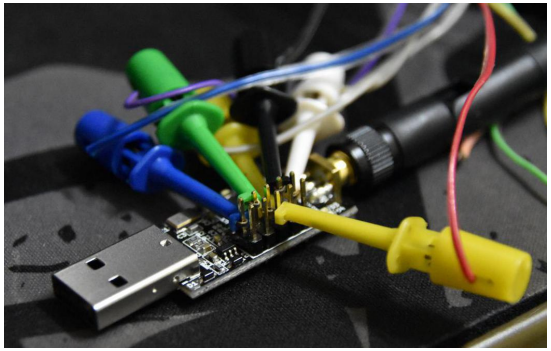
Instructions:

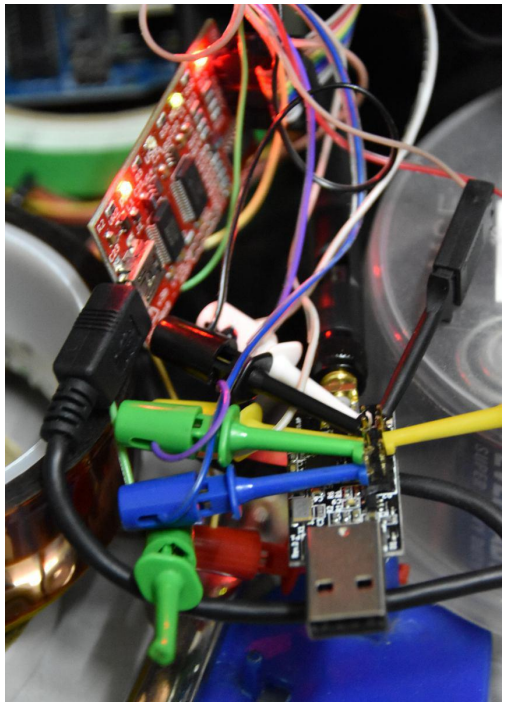
1. Solder a 2×5 pin header onto the programming port of the crazyradio. There's an unpopulated footprint already there for you.
2. Connect the crazyradio to your Bus Pirate using the table below (also noted in the script and readme on git)

```
Bus Pirate       CrazyRadio
============================
MOSI ()      ->  MOSI  (6)
MISO ()      ->  MISO  (8)
SCK  ()      ->  SCK   (4)
CS   ()      ->  CS    (10)
AUX  ()      ->  PROG  (2)
3V3  ()      ->  3V3   (5)
GND  ()      ->  GND   (9)
```

3. Run the script: perl ./flasher.pl -input ./cradio-0.51.bin -device [serial device]
4. Wait till you see lots of hex addresses crawling up your screen. Your device is programming.
5. Go make a sandwich or have a beer (or both).







- Crazyradio comes with breakout pin
- Solder a 2 x 5 Pin into crazy radio
- “Clipped” in Bus Pirate accordingly
- Beware of crazyradio breakout pin sequence

Problem 1: Boot Loader Missing

Re: Bus Pirate script to recover bricked radio

by **arnaud** » Sun Jun 29, 2014 10:57 am

arnaud
Site Admin
Posts: 434
Joined: Tue Feb 06, 2007 12:36 pm

Hi Everdoubtful,

Apparently the script has erased the entire chip including the nrf usb bootloader, which is bad.

To get the radio to work again flash the normal firmware, the latest version can be download from there
https://bitbucket.org/bitcraze/crazyrad ... /downloads
Otherwise for a more permanent solution I uploaded a bin version of the bootloader there http://files1.bitcraze.se/dl/boot24lu1p-f32.bin. Until the perl script is fixed this is 32K so it will take some time to flash.

I don't have access to a buspirate right now but I will look at it tomorrow to fix the script.

/Arnaud

Re: Bus Pirate script to recover bricked radio

by **koolatron** » Mon Jul 28, 2014 9:02 pm

koolatron
Posts: 3
Joined: Sat Jun 01, 2013 5:08 am

Yes, the script I wrote executes ERASE_ALL so it is intended only to flash images that contain a copy of the bootloader. It was never intended to take a truncated "jump to bootloader" bin.

- Due to the “PA” flashed in to the NON “PA”, it overwrites the boot loader
- Almost broken Perl script not able to execute completely
- ERASE_ALL makes it all worse
- Info: https://forum.bitcraze.io/viewtopic.php?t=323

The Boot Loader

Re: Bus Pirate script to recover bricked radio

by **arnaud** » Sun Jun 29, 2014 10:57 am

arnaud
Site Admin

Posts: 434
Joined: Tue Feb 06, 2007 12:36 pm

Hi Everdoubtful,

Apparently the script has erased the entire chip including the nrf usb bootloader, which is bad.

To get the radio to work again flash the normal firmware, the latest version can be download from there
https://bitbucket.org/bitcraze/crazyrad ... /downloads
Otherwise for a more permanent solution I uploaded a bin version of the bootloader there http://files1.bitcraze.se/dl/boot24lu1p-f32.bin. Until the perl script is fixed this is 32K so it will take some time to flash.

I don't have access to a buspirate right now but I will look at it tomorrow to fix the script.

/Arnaud

Re: Bus Pirate script to recover bricked radio

by **koolatron** » Mon Jul 28, 2014 9:02 pm

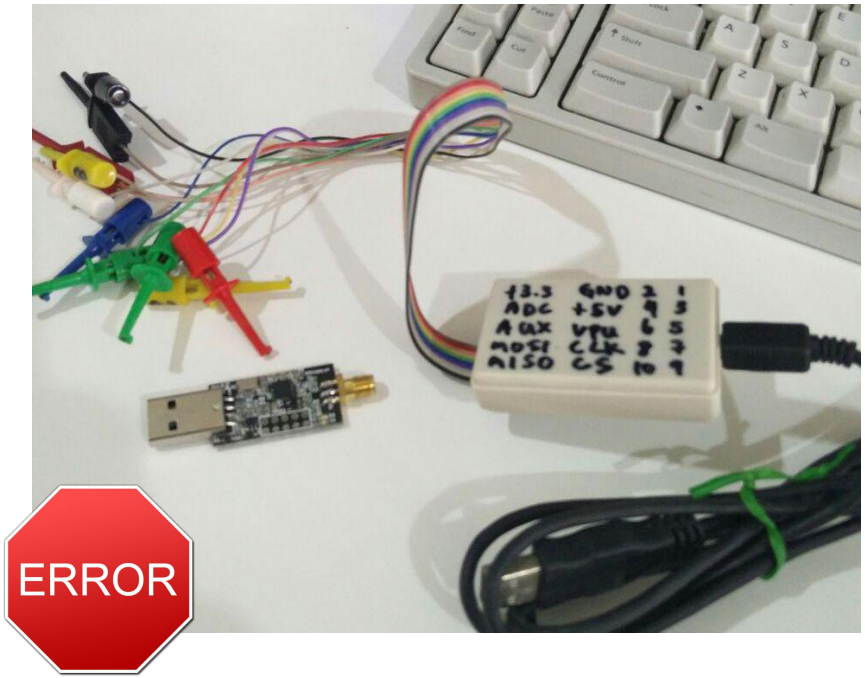
koolatron

Posts: 3
Joined: Sat Jun 01, 2013 5:08 am

Yes, the script I wrote executes ERASE_ALL so it is intended only to flash images that contain a copy of the bootloader. It was never intended to take a truncated "jump to bootloader" bin.

- The possible way is, flash the boot loader
- Once completed, flash the crazyradio firmware
- Boot Loder: https://github.com/xwings/tuya

The Final Error



- # git clone https://github.com/RFStorm/mousejack.git
- # cd mousejack
- # make
- Flash the firmware into crazyradio
- Almost working Perl script not working, almost working and almost broken

The “Broken” Perl Script

```perl
use strict;
use feature 'say';
use Getopt::Long;
use Device::SerialPort;
use Time::HiRes qw/usleep/;

use constant {
    WREN        => "\x06",
    WRDIS       => "\x04",
    RDSR        => "\x05",
    WRSR        => "\x01",
    READ        => "\x03",
    PROGRAM     => "\x02",
    ERASE_PAGE  => "\x52",
    ERASE_ALL   => "\x62",
    RDFPCR      => "\x89",
    RDISMB      => "\x85",
    ENDEBUG     => "\x86",
    RDYN        => "\x10",
    FLASH_LEN   => 32768,

    BP_CS       => "\x01",
    BP_AUX      => "\x02",
    BP_PULLUP   => "\x04",
    BP_POWER    => "\x08",
};

my %opts;
my $port;
my $time = 500;
my $status_byte;
my $return;

if (!&GetOptions(\%opts,
    'input=s',
    'device=s',
) || ( !$opts{input} && !$opts{device} ) ) {
    die "Please specify both -input <input_file.bin> and -device <Bus Pirate devnode>";
}

$port = new Device::SerialPort( $opts{device} );

# Setup serial

$port->baudrate(115200);
$port->parity("none");
$port->databits(8);
$port->stopbits(1);
$port->buffers(1,1);
$port->write_settings || undef $port;

die "Unable to write settings to serial port." unless $port;

# Setup BP
say "Entering raw bitbang mode...";
    while ( ( $port->read( 5 ) ne "BBIO1" ) && --$time ) {
    $port->write( "\x00" );
    usleep( 20000 );
}
die "Unable to enter raw bitbang mode!" unless $time;
```

```perl
use strict;
use feature 'say';
use Getopt::Long;
use Device::SerialPort;
use Time::HiRes qw/usleep/;

use constant {
    WREN        => "\x06",
    WRDIS       => "\x04",
    RDSR        => "\x05",
    WRSR        => "\x01",
    READ        => "\x03",
    PROGRAM     => "\x02",
    ERASE_PAGE  => "\x52",
    ERASE_ALL   => "\x62",
    RDFPCR      => "\x89",
    RDISMB      => "\x85",
    ENDEBUG     => "\x86",
    RDYN        => "\x10",
    FLASH_LEN   => 32768,

    BP_CS       => "\x01",
    BP_AUX      => "\x02",
    BP_PULLUP   => "\x04",
    BP_POWER    => "\x08",
};

my %opts;
my $port;
my $time = 500;
my $status_byte;
my $return;

if (!&GetOptions(\%opts,
    'input=s',
    'device=s',
) || ( !$opts{input} && !$opts{device} ) ) {
    die "Please specify both -input <input_file.bin> and -device <Bus Pirate de
}

$port = new Device::SerialPort( $opts{device} );

# Setup serial

$port->baudrate(115200);
$port->parity("none");
$port->databits(8);
$port->stopbits(1);
$port->buffers(1,1);
$port->write_settings || undef $port;

die "Unable to write settings to serial port." unless $port;

# Setup BP
say "Entering raw bitbang mode...";
    while ( ( $port->read( 5 ) ne "BBIO1" ) && --$time ) {
    usleep( 40000 );
}
die "Unable to enter raw bitbang mode!" unless $time;
```

Branch: master ▾ tuya / mousejack / klks_buspirate /

xwings update logictech and elks code

..

pyBusPirateLite	update logictech and elks code
7800_bootloader.hex	update logictech and elks code
klks_commoncode.py	update logictech and elks code
klks_readinfopage.py	update logictech and elks code
klks_readmainblock.py	update logictech and elks code
klks_writebootloader.py	update logictech and elks code
klks_writeinfopage.py	update logictech and elks code
klks_writemainblock.py	update logictech and elks code

- The Perl script is broken by default under VM
- Replace all usleep(20000) to usleep(40000)
- Completely Re-implemented in python: https://github.com/xwings/tuya
- Did I mention within two hours

Re-Flash

Boot Loader

```
root@kali:~/buspirate_nrf24lu1p# perl flasher.pl -i boot24lu1p-f32.bin -device /
dev/ttyUSB0
Entering raw bitbang mode...
Entering binary SPI mode...
Configuring peripherals...
Configuring SPI...
Enabling programming...
Reading status byte...
Status: 20
Erasing chip...
Enabling programming...
Reading status byte...
Status: 20
Programming device...
0000 : 0278
0002 : 0000
```

Firmware

```
root@kali:~/buspirate_nrf24lu1p# perl flasher.pl -input cradio-0.51.bin -device /dev/ttyUSB0
Entering raw bitbang mode...
Entering binary SPI mode...
Configuring peripherals...
Configuring SPI...
Enabling programming...
Reading status byte...
Status: 20
Erasing chip...
Enabling programming...
Reading status byte...
Status: 20
Programming device...
0000 : 0200
0002 : 6b32
0004 : 0000
```

- Two hours for the bootloader
- Two hours for the crazyradio firmware
- Two hours for the mousejack firmware
- Ok, Maybe two hours. I went out after the flash started

It Works

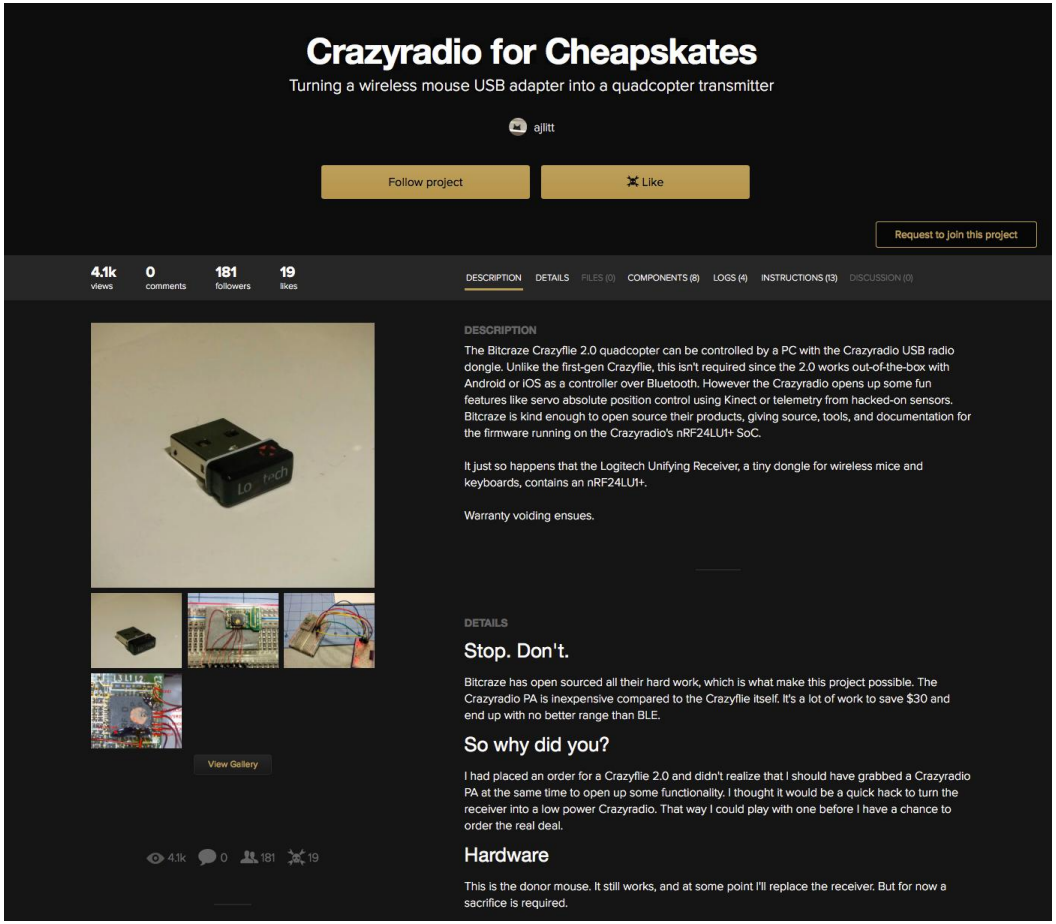
```
[  416.993066] usb 1-2.2: new full-speed USB device number 7 using uhci_hcd
[  417.089596] usb 1-2.2: New USB device found, idVendor=1915, idProduct=0102
[  417.089599] usb 1-2.2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[  417.089600] usb 1-2.2: Product: Research Firmware
[  417.089601] usb 1-2.2: Manufacturer: RFStorm
```

```
(15)# python ./nrf24-scanner.py
[2016-03-24 21:20:07.388]  32   0  72:E4:
[2016-03-24 21:20:07.425]  32   0  72:E4:
[2016-03-24 21:20:07.458]  32  10  72:E4:  00:C2:00:00:02:D0:FF:00:00:6D
[2016-03-24 21:20:32.988]  32   5  72:E4:  00:40:00:6E:52
```

```
(21)# python ./nrf24-sniffer.py -a 72:E4
[2016-03-24 21:23:08.242]  32   5  72:E4  00:40:00:6E:52
[2016-03-24 21:23:08.335]  32   5  72:E4  00:40:00:6E:52
[2016-03-24 21:23:08.427]  32   5  72:E4  00:40:00:6E:52
[2016-03-24 21:23:08.521]  32  10  72:E4  00:C2:00:00:FA:0F:00:00:00:35
[2016-03-24 21:23:08.529]  32  10  72:E4  00:C2:00:00:F4:0F:00:00:00:3B
[2016-03-24 21:23:08.537]  32  10  72:E4  00:C2:00:00:F0:0F:00:00:00:3F
[2016-03-24 21:23:08.544]  32  10  72:E4  00:C2:00:00:F4:FF:FF:00:00:4C
[2016-03-24 21:23:08.552]  32  10  72:E4  00:C2:00:00:F5:DF:FF:00:00:6B
[2016-03-24 21:23:08.559]  32  10  72:E4  00:C2:00:00:FA:EF:FF:00:00:56
[2016-03-24 21:23:08.569]  32  10  72:E4  00:C2:00:00:FE:FF:FF:00:00:42
[2016-03-24 21:23:08.580]  32  10  72:E4  00:C2:00:00:FE:FF:FF:00:00:42
[2016-03-24 21:23:08.593]  32  10  72:E4  00:4F:00:00:6E:00:00:00:00:43
[2016-03-24 21:23:08.600]  32   5  72:E4  00:40:00:6E:52
[2016-03-24 21:23:08.693]  32   5  72:E4  00:40:00:6E:52
[2016-03-24 21:23:08.732]  32  10  72:E4  00:C2:00:00:00:10:00:00:00:2E
[2016-03-24 21:23:08.739]  32  10  72:E4  00:4F:00:00:6E:00:00:00:00:43
[2016-03-24 21:23:08.756]  32  10  72:E4  00:C2:00:00:01:20:00:00:00:1D
[2016-03-24 21:23:08.763]  32  10  72:E4  00:4F:00:00:6E:00:00:00:00:43
```

End is Another Start

Google Found This



Crazyradio for Cheapskates

Turning a wireless mouse USB adapter into a quadcopter transmitter

ajlitt

Follow project | Like

Request to join this project

4.1k views | 0 comments | 181 followers | 19 likes

DESCRIPTION | DETAILS | FILES (0) | COMPONENTS (8) | LOGS (4) | INSTRUCTIONS (13) | DISCUSSION (0)

View Gallery

4.1k 0 181 19

DESCRIPTION

The Bitcraze Crazyflie 2.0 quadcopter can be controlled by a PC with the Crazyradio USB radio dongle. Unlike the first-gen Crazyflie, this isn't required since the 2.0 works out-of-the-box with Android or iOS as a controller over Bluetooth. However the Crazyradio opens up some fun features like servo absolute position control using Kinect or telemetry from hacked-on sensors. Bitcraze is kind enough to open source their products, giving source, tools, and documentation for the firmware running on the Crazyradio's nRF24LU1+ SoC.

It just so happens that the Logitech Unifying Receiver, a tiny dongle for wireless mice and keyboards, contains an nRF24LU1+.

Warranty voiding ensues.

DETAILS

Stop. Don't.

Bitcraze has open sourced all their hard work, which is what make this project possible. The Crazyradio PA is inexpensive compared to the Crazyflie itself. It's a lot of work to save $30 and end up with no better range than BLE.

So why did you?

I had placed an order for a Crazyflie 2.0 and didn't realize that I should have grabbed a Crazyradio PA at the same time to open up some functionality. I thought it would be a quick hack to turn the receiver into a low power Crazyradio. That way I could play with one before I have a chance to order the real deal.

Hardware

This is the donor mouse. It still works, and at some point I'll replace the receiver. But for now a sacrifice is required.

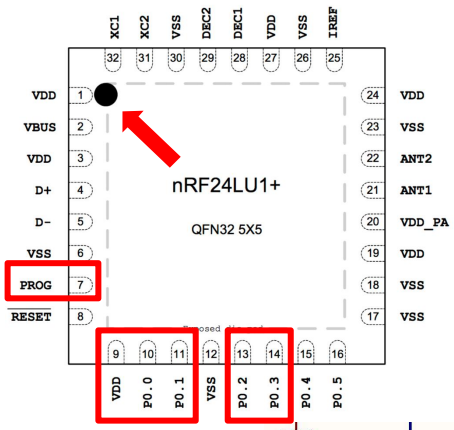
- We found someone actually trying to fly crazyflies with Logitech unify dongle
- If Logitech Unify dongle compatible with crazyradio firmware, it means ….
- https://hackaday.io/project/6741-crazyradio-for-cheapskates

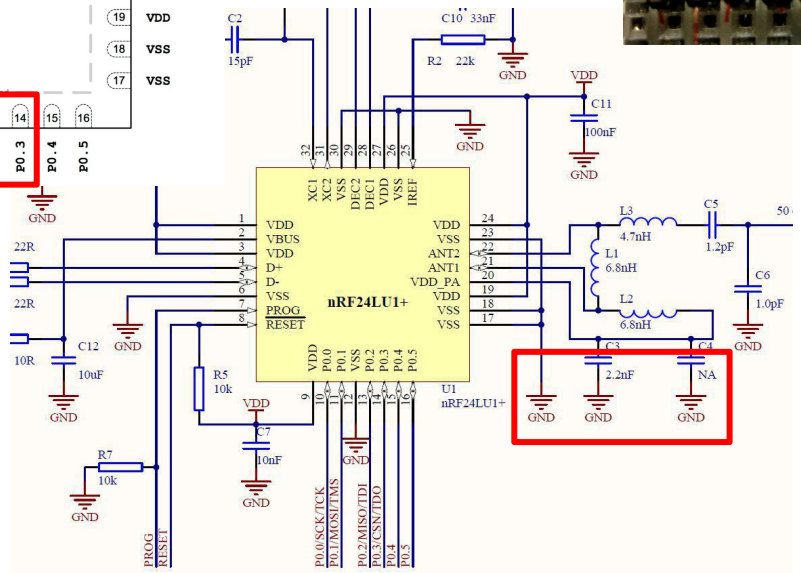
What is Logitech Unifying Receiver

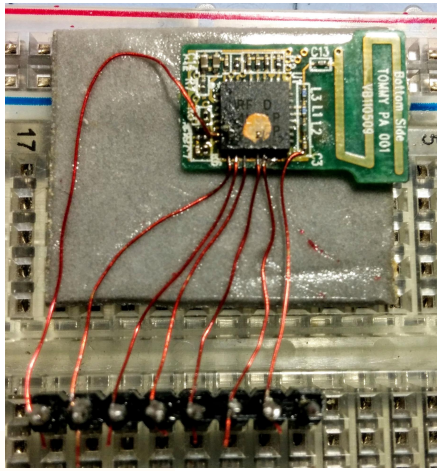
- One for all, all for one
- 25 RMB at taobao

Identifying the Pins

```
#!/usr/bin/perl -w

# Simple perl script to drive the Bus Pirate and unbrick your CrazyRadio dongle.
# Adapted (sorta) from the Bus Pirate example script and mbed NRF24LU1+ flasher projects:
#  http://code.google.com/p/the-bus-pirate/source/browse/trunk/scripts/SPIeeprom.pl
#  http://mbed.org/users/mux/code/nrflash
#
# This script uses the aux output on the Bus Pirate as the PROG pin on the CrazyRadio's NRF24LU1+ chip.
#
# Electrical connections are as follows:
#
#  Bus Pirate        CrazyRadio
#  ==========================
#  MOSI ()      ->  MOSI   (6)
#  MISO ()      ->  MISO   (8)
#  SCK  ()      ->  SCK    (4)
#  CS   ()      ->  CS     (10)
#  AUX  ()      ->  PROG   (2)
#  3V3  ()      ->  3V3    (5)
#  GND  ()      ->  GND    (9)

use strict;
use feature 'say';
use Getopt::Long;
use Device::SerialPort;
use Time::HiRes qw/usleep/;

use constant {
    WREN        => "\x06",
    WRDIS       => "\x04",
    RDSR        => "\x05",
    WRSR        => "\x01",
    READ        => "\x03",
    PROGRAM     => "\x02",
    ERASE_PAGE  => "\x52",
    ERASE_ALL   => "\x62",
    RDFPCR      => "\x89",
    RDISMB      => "\x85",
    ENDEBUG     => "\x86",
    RDYN        => "\x10",
    FLASH_LEN   => 32768,

    BP_CS       => "\x01",
    BP_AUX      => "\x02",
    BP_PULLUP   => "\x04",
    BP_POWER    => "\x08",
};

my %opts;
my $port;
my $time = 500;
my $status_byte;
my $return;
```

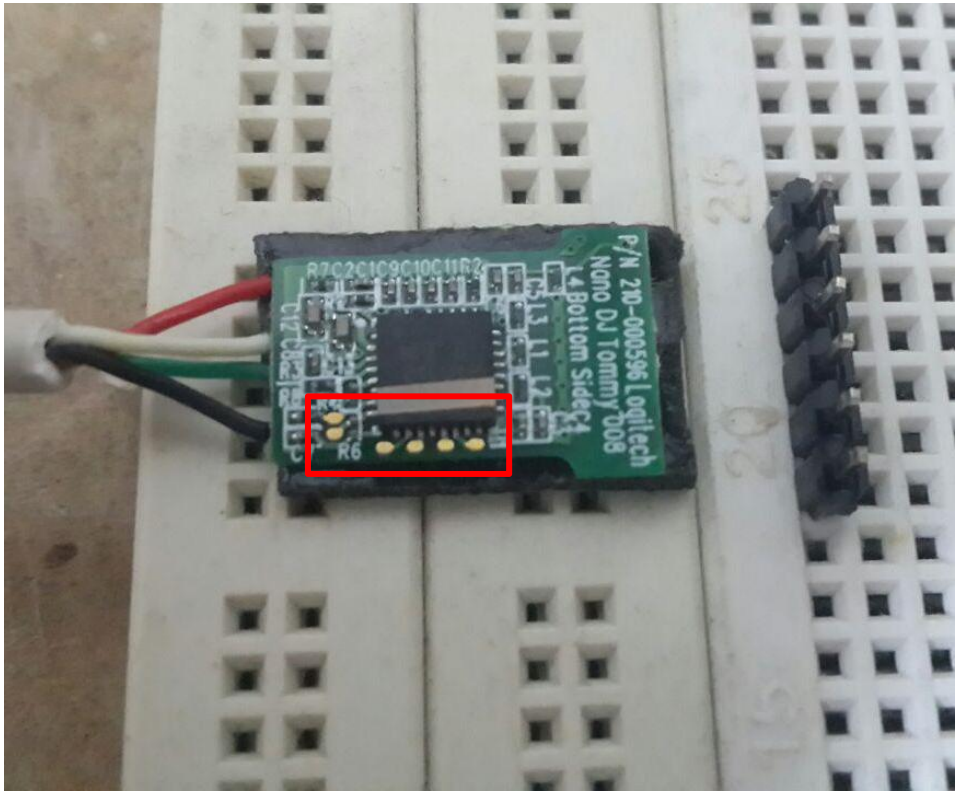






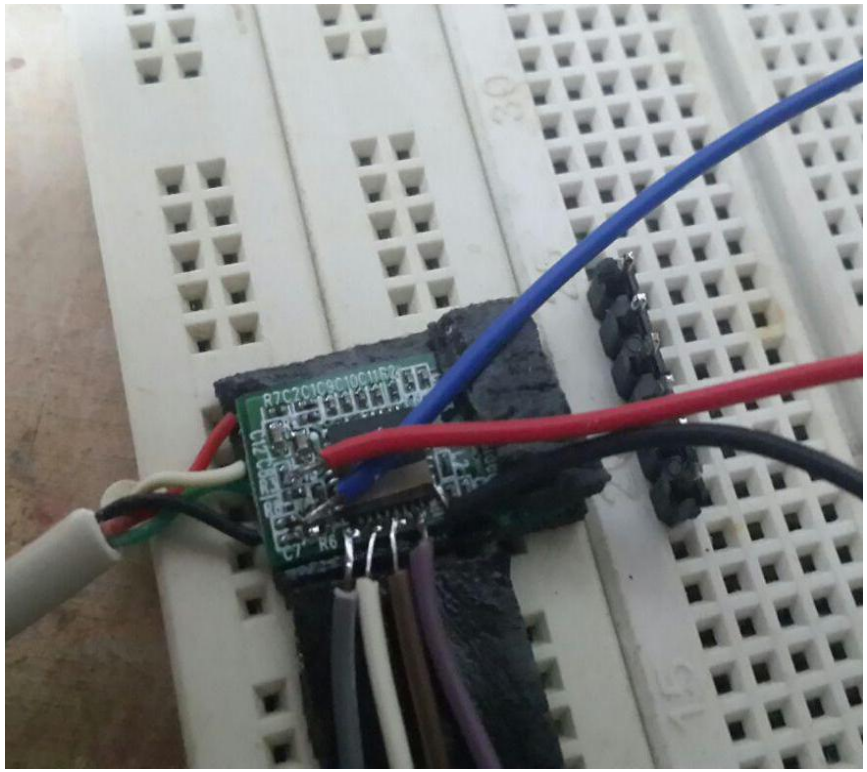
- MOSI - Pin 11
- MISO - Pin 13
- SCK - Pin 10
- CS - PIN 14
- AUX – Pin 7
- 3V3 - Pin 1
- GND – Any GND

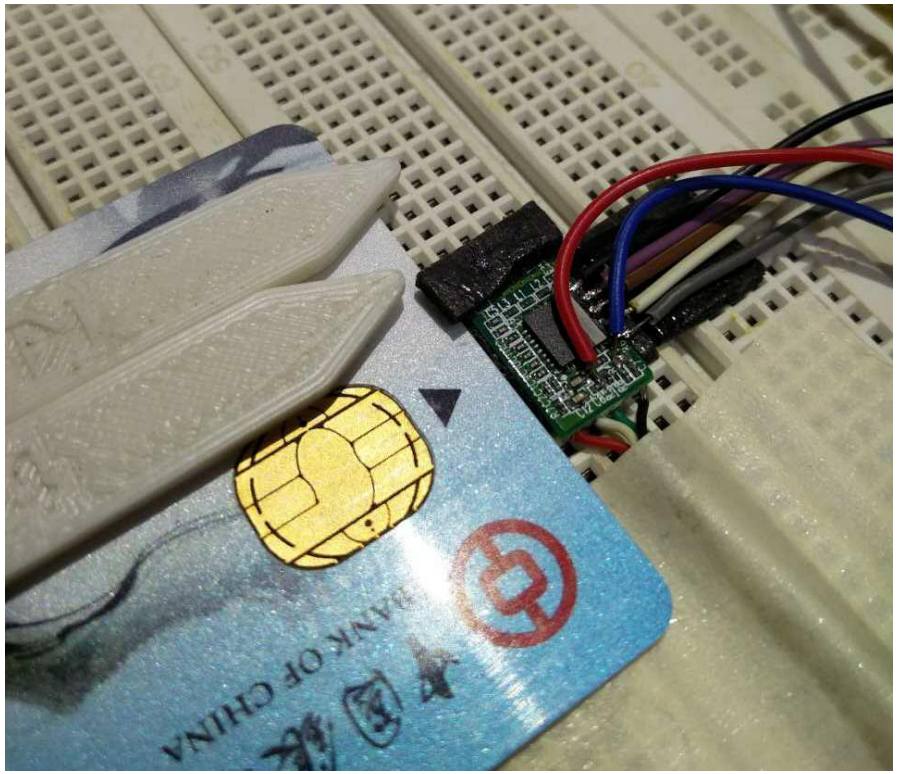
What Is in Logitech Unifying Dongle



- Open up the casing
- It comes with breakout PINS !
- Find the GRD
- ULTRA STABLE HAND

Soldering





- Breakout Pin save the world
- Soldering all the Pin accordingly
- Connects to BUS Pirate
- Start Flashing the boot loader
- Flash MouseJack firmware

Re-Flash

Boot loader

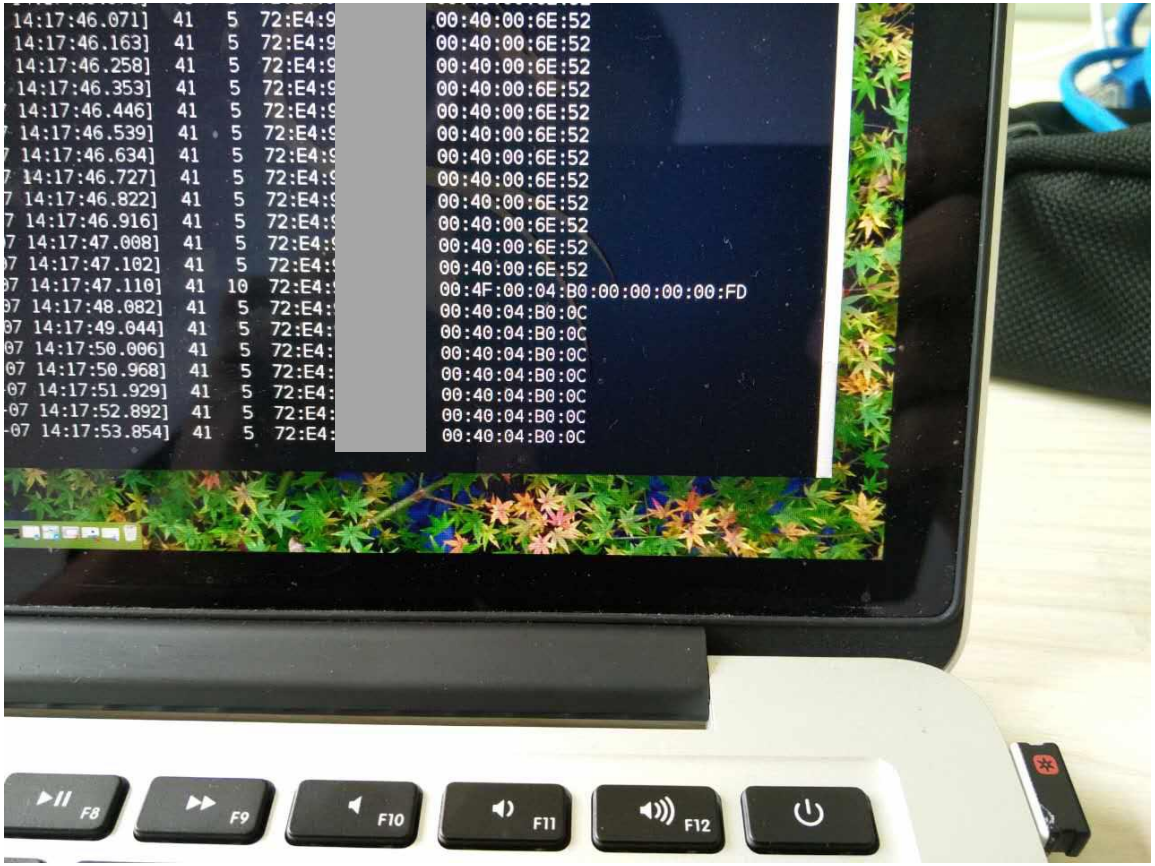
```
root@kali:~/buspirate_nrf24lu1p# perl flasher.pl -i boot24lu1p-f32.bin -device /
dev/ttyUSB0
Entering raw bitbang mode...
Entering binary SPI mode...
Configuring peripherals...
Configuring SPI...
Enabling programming...
Reading status byte...
Status: 20
Erasing chip...
Enabling programming...
Reading status byte...
Status: 20
Programming device...
0000 : 0278
0002 : 0000
```

firmware

```
root@kali:~/buspirate_nrf24lu1p# perl flasher.pl -input cradio-0.51.bin -device /dev/ttyUSB0
Entering raw bitbang mode...
Entering binary SPI mode...
Configuring peripherals...
Configuring SPI...
Enabling programming...
Reading status byte...
Status: 20
Erasing chip...
Enabling programming...
Reading status byte...
Status: 20
Programming device...
0000 : 0200
0002 : 6b32
0004 : 0000
```

- Two hours for the bootloader
- Two hours for the crazyradio firmware
- Two hours for the mousejack firmware
- Again, maybe not two hours

It Works



- Connect to PC
- Run the scanner - works
- Run the sniffer - works
- Replay works

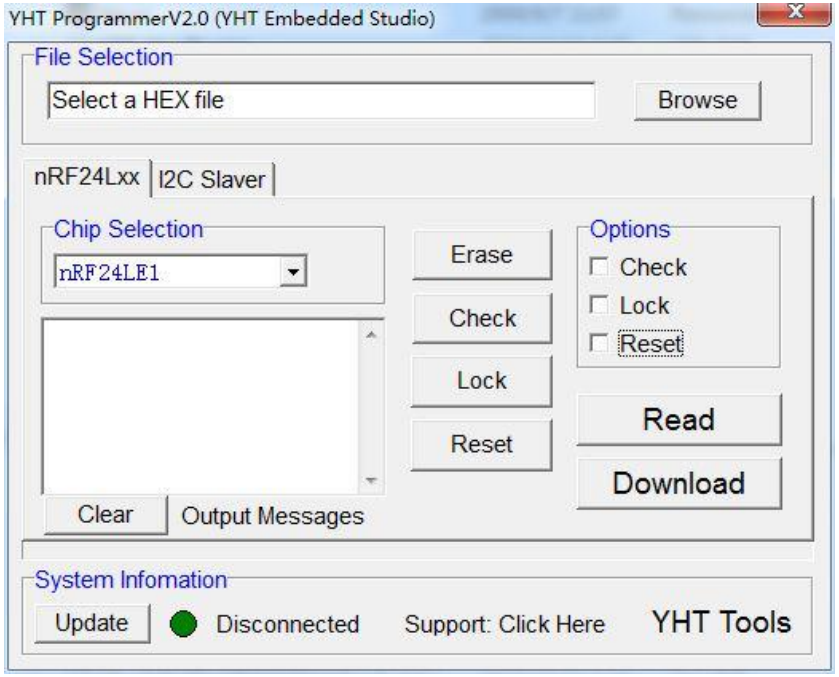
One Problem Left

Bus Pirate

- Bus Pirate is expensive
- Back to back order
- Long waiting time
- EXPENSIVE !!!

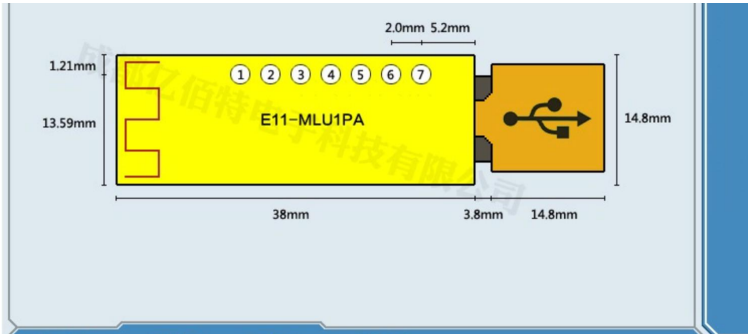
We (Wo) Love (Da) China (TianChao)

First Buy – PA Unit



- Identify all the 7 Pins from the programmer and USB dongle
- Soldering is needed
- “Rainbow” connector is needed, both female
- Single row pin is needed/Pogo Pin
- 119 RMB + 40 RMB = 159 RMB

Pin Mapping



引脚序号	引脚名称	引脚方向	引脚用途
01	GND		地线，连接到电源参考地
02	CSN	输入	SPI片选信号
	P0.3	输入/输出	单片机GPIO
03	MISO	输出	SPI数据输出
	P0.2	输入/输出	单片机GPIO
	TXD	输出	UART输出，详见nRF24LU1P数据手册
04	MOSI	输入	SPI数据输入
	P0.1	输入/输出	单片机GPIO
	RXD	输入	UART输入，详见nRF24LU1P数据手册
05	SCK	输入	SPI总线时钟
	P0.0	输入/输出	单片机GPIO
06	PROG	输入	编程使能，平时悬空即可
07	VBUS		电源参考 +5V

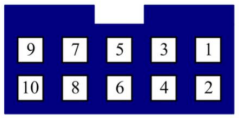


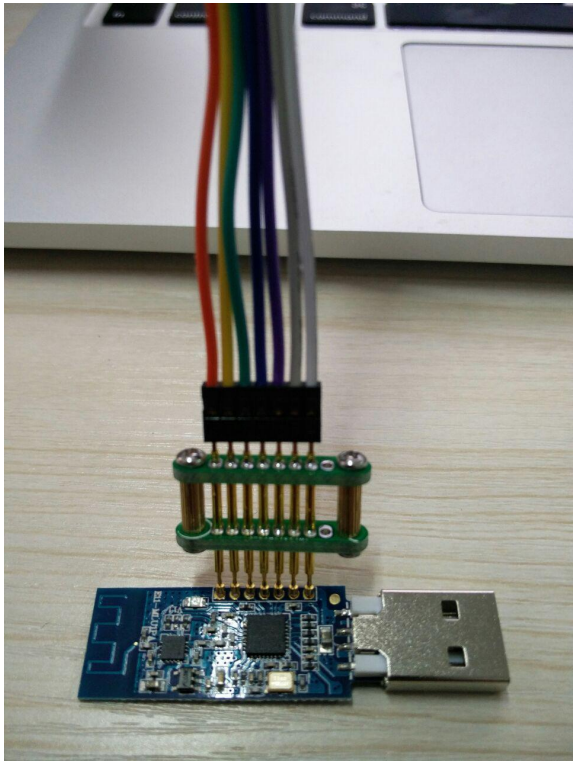
图 2：编程接口示意图

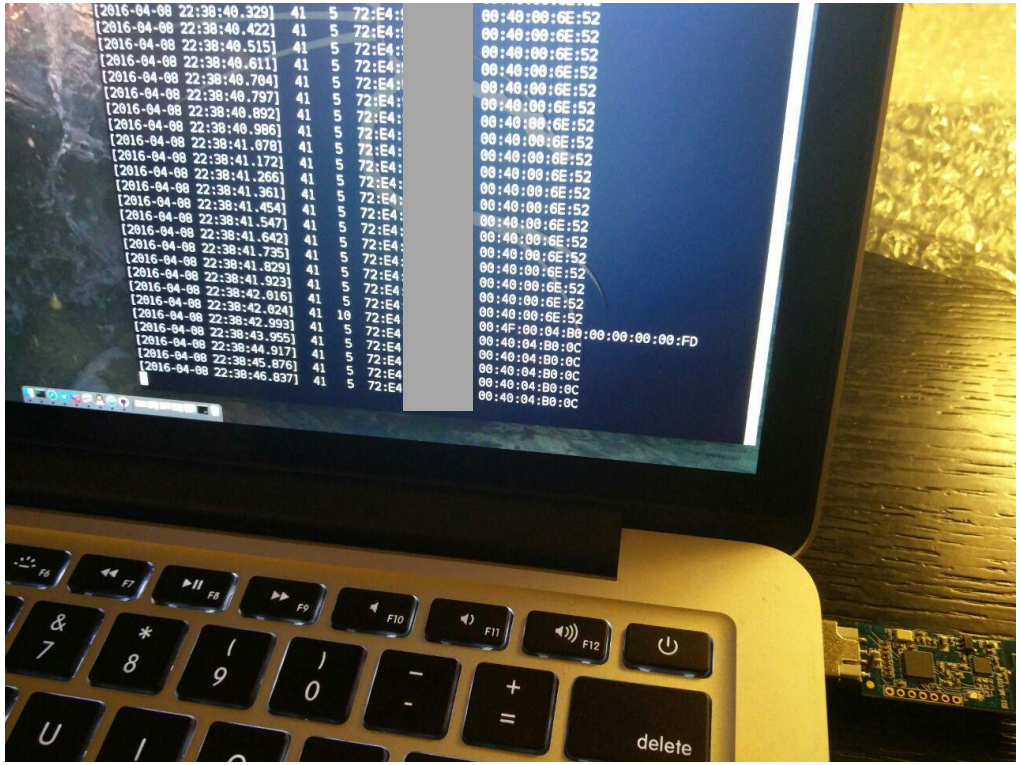
表 1：编程接口管脚说明

序号	名称	描述
1.	+5V	5V 电源正
2.	+3.3V	3.3V 电源正
3.	RESET	复位信号
4.	NSS	SPI 片选信号
5.	MISO	SPI 主入从出
6.	PROG	芯片编程使能：高有效
7.	MOSI	SPI 主出从入
8.	SCK	SPI 时钟
9.	NC	未连接
10.	GND	GND

- Flash with pogo pin
- Comes with Software
- All pins are clear, except CSN needs to map to NSS
- PA Unit

Flash



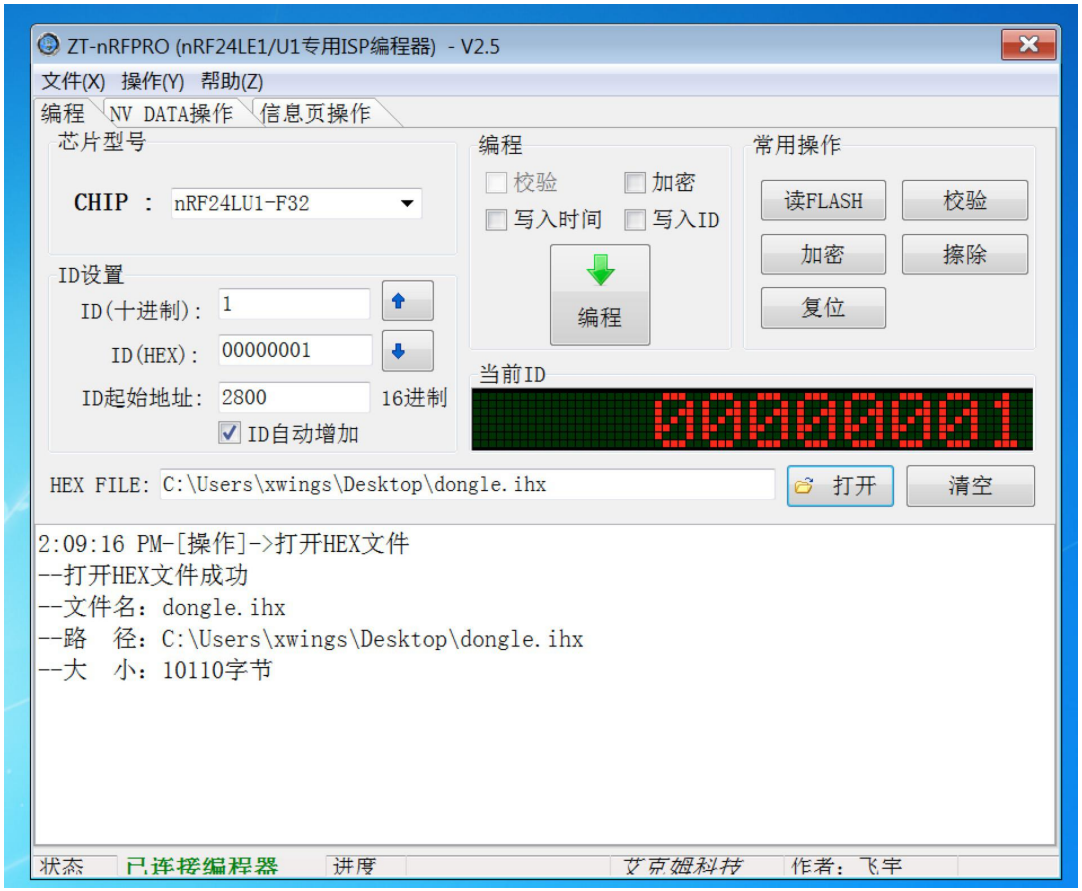


- Connect to PC
- Align Pogo Pins
- Flash
- Run the scanner - works
- Run the sniffer - works

Second Buy – Non PA Unit

- No Soldering required
- Comes with Software
- 99 RMB One complete set, just like buying a mac
- This should be the NON-PA unit

Flash



- Computer USB -> Programmer -> Connector -> Breakout Pins
- 5 Seconds Flashing
- End before you start
- Done !!!

It Works

- Run the scanner - works
- Run the sniffer - works

Conclusion

Cheap Way, Easy Way = Good Way

- Made in China
- Programmer can flash crazyradio and Logitech Unifying dongle
- Easy to get. Stock always available
- Cheap and cheap and cheap

One More Thing

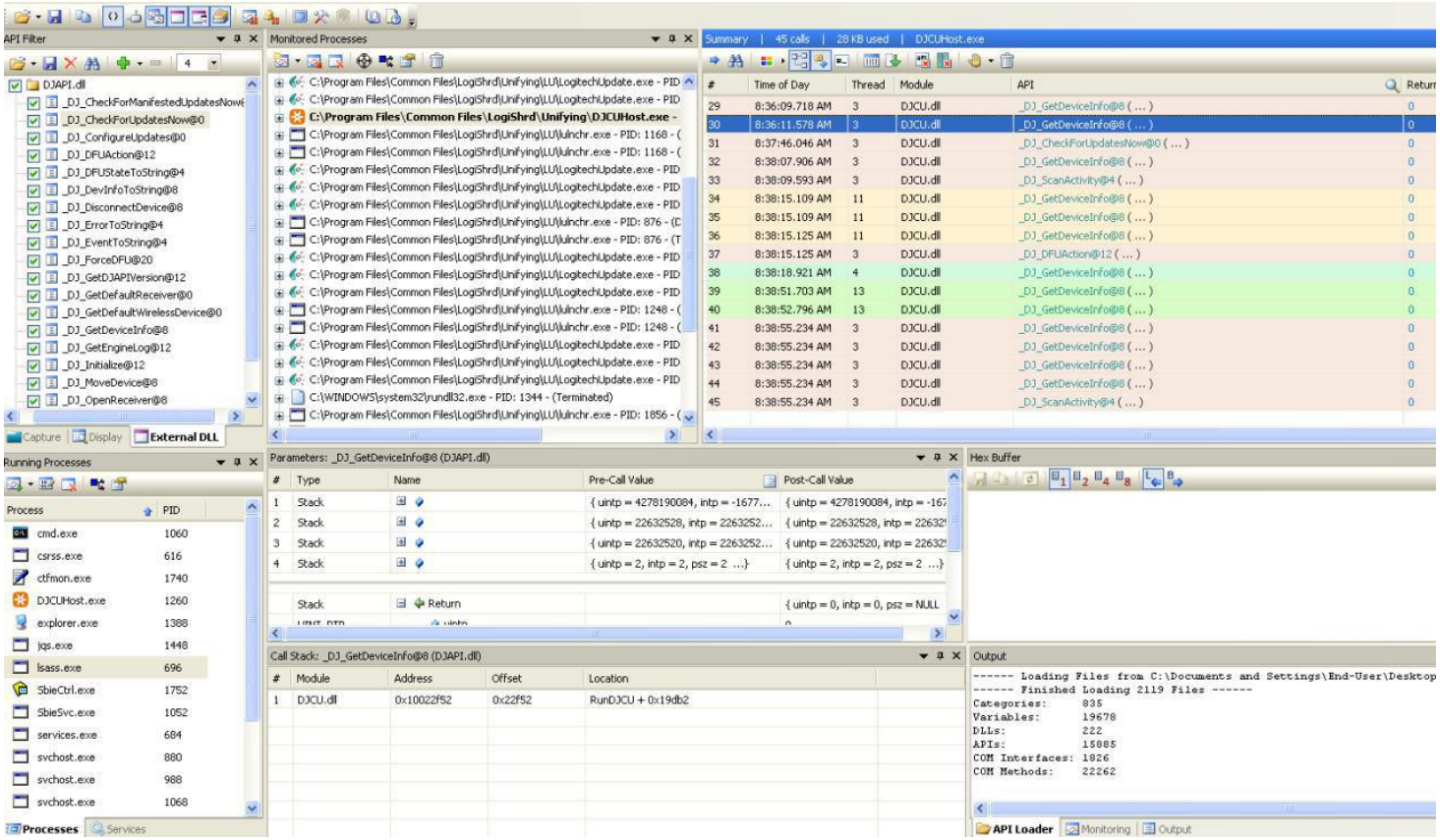
What Is Missing

- Is there really why it call MouseJack?
- Only Mouse at the moment?
- Possible to hijack a keyboard?

Having Fun with Logitech Keyboard

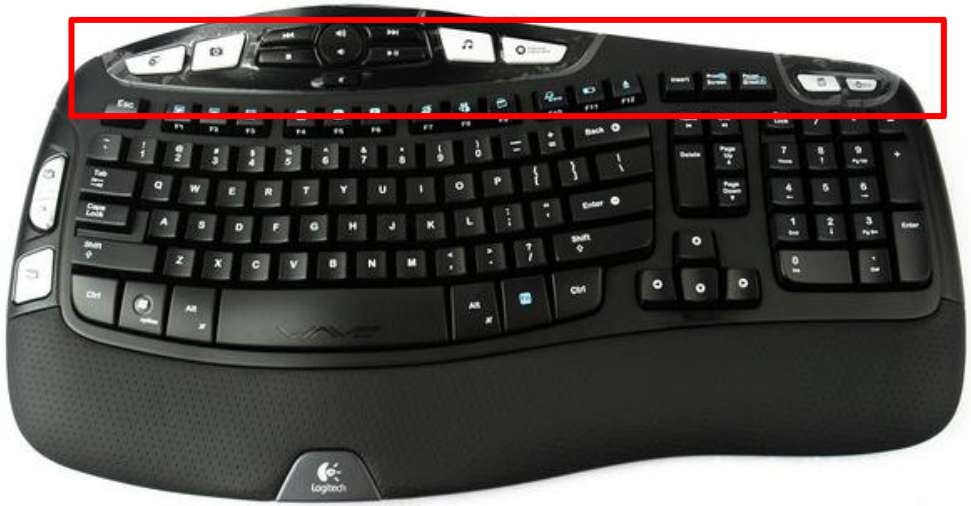
- Most popular brand, Logitech
- Lets see what is in Logitech Keyboard

What We Know



- AES 128bit encryption between keyboard and dongle
- Able to dump some functions
- Few projects doing Logitech Unifying Keyboard, such as solaar. http://pwr.github.io/Solaar
- Time is too limited and nothing much able to capture from the trace

Some Info on Wireless Keyboard



- Most of the multimedia key seems to be not being encrypted
- Not enough to encrypt all the keys ?

What If, Keyboard Is Not Available

- No one will bring a wireless keyboard outdoor
- Send in unencrypted keystroke to mouse dongle? Yes, it works
- Sending encrypted keystroke using unencrypted method. Example, brute force?
- Or Presenter ?

Dumping the Firmware

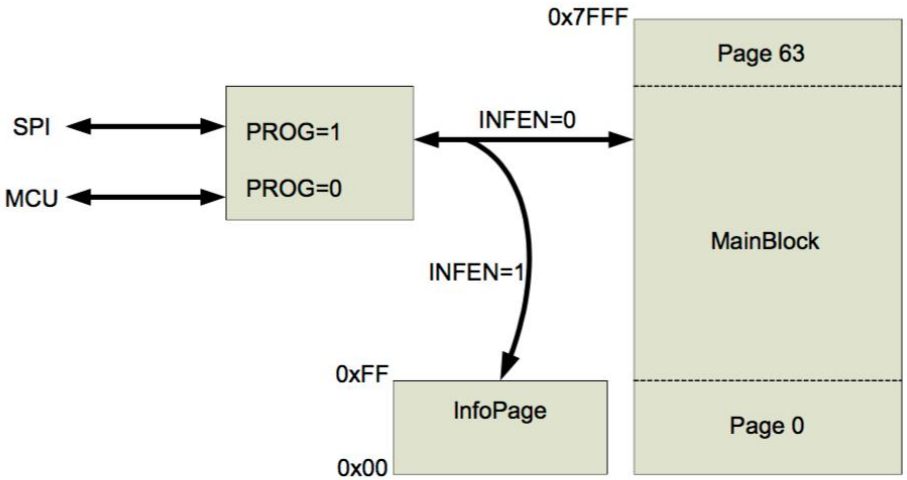


Figure 62. Flash memory block diagram

- Almost Not Possible
- InfoPage Readback blocking, 0x22
- MainBlock Readback blocking, 0x23
- Only can perform complete rewrite

Getting the Firmware

Reply | Topic Options | Message Listing | Previous Topic | Next Topic

ModeratorTeam Logitech
Moderator
Posts: 227
Registered: 08-25-2010

Logitech Response to Unifying Receiver Research Findings [Edited]

Options

02-23-2016 09:10 AM - edited 02-24-2016 03:16 PM

You may have read or heard that researchers from Bastille Security found a potential vulnerability in Logitech's Unifying receiver. The Unifying receiver allows you to connect multiple compatible keyboards and mice to a laptop or desktop computer with a single USB receiver.

Bastille Security approached us regarding their work. We have been in regular communication with them since and together have discussed their findings.

Bastille Security identified the vulnerability in a controlled, experimental environment. The vulnerability would be complex to replicate and would require physical proximity to the target. It is therefore a difficult and unlikely path of attack.

We have nonetheless taken Bastille Security's work seriously and developed a firmware fix. If you have concerns, and would like to ensure this vulnerability is eliminated, you can follow these steps:

1. Download and install **Unifying Software**.
2. To check the firmware version on your Unifying receiver, go to **Unifying Software → Advanced**, and then select the **Unifying Receiver**.
3. The version of the firmware is listed in the right pane.
   - If the firmware version is in 0**12**.xxx.000xx format, download and save RQR_012_005_00028.exe through the following link: **http://logt.ly/0222**
   - If the firmware version is in 0**24**.xxx.000xx format, download and save RQR_024_003_00027.exe through the following link: **http://logt.ly/0224**
4. Run the downloaded firmware package.
5. Open **Unifying Software → Advanced**, then select the **Unifying Receiver.**
6. In the right pane, click on **Update Firmware** and wait until the firmware update is complete.

Note: To have all the features working correctly after updating the firmware, please ensure that you have the latest version of **SetPoint** and/or **Options** software that supports your device.

Logitech's Unifying technology was launched in 2007 and has been used by millions of our consumers since. To our knowledge, we have never been contacted by any customer with such an issue related to this potential vulnerability.

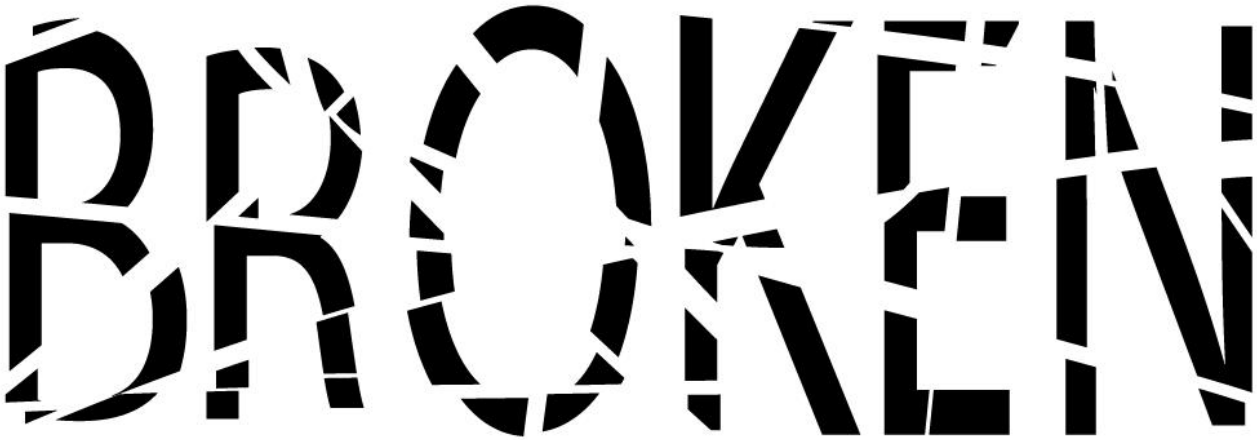
- Download
- Simple RE
- Got the firmware in HEX
- https://github.com/xwings/tuya

Intel 8051

```
code:0000145D E5 3A                    mov     A, 0x3A
code:0000145F 70 12                    jnz     code_1473
code:00001461 75 76 05                 mov     0x76, #5
code:00001464 75 77 01                 mov     0x77, #1
code:00001467 90 83 7E                 mov     DPTR, #0x837E
code:0000146A E0                       movx    A, @DPTR
code:0000146B 54 FE                    anl     A, #0xFE
code:0000146D F0                       movx    @DPTR, A
code:0000146E 54 FD                    anl     A, #0xFD
code:00001470 F0                       movx    @DPTR, A
code:00001471 01 30                    ajmp    code_1030
code:00001473                  ; ---------------------------------------------------------------
code:00001473
code:00001473
code:00001473                  code_1473:                      ; CODE XREF: code_143A+25↑j
code:00001473 E5 3A                    mov     A, 0x3A
code:00001475 B4 01 0F                 cjne    A, #1, code_1487
code:00001478 75 76 05                 mov     0x76, #5
code:0000147B 75 77 02                 mov     0x77, #2
code:0000147E 90 83 7E                 mov     DPTR, #0x837E
code:00001481 E0                       movx    A, @DPTR
code:00001482 54 FD                    anl     A, #0xFD
code:00001484 F0                       movx    @DPTR, A
code:00001485 21 38                    ajmp    code_1138
code:00001487                  ; ---------------------------------------------------------------
code:00001487
code:00001487
code:00001487                  code_1487:                      ; CODE XREF: code_143A+3B↑j
code:00001487 E5 3A                    mov     A, 0x3A
code:00001489 B4 02 0F                 cjne    A, #2, code_149B
code:0000148C 75 76 05                 mov     0x76, #5
code:0000148F 75 77 03                 mov     0x77, #3
code:00001492 90 83 7E                 mov     DPTR, #0x837E
code:00001495 E0                       movx    A, @DPTR
code:00001496 44 02                    orl     A, #2
code:00001498 F0                       movx    @DPTR, A
code:00001499 21 86                    ajmp    code_1186
code:0000149B                  ; ---------------------------------------------------------------
code:0000149B
code:0000149B                  code_149B:                      ; CODE XREF: code_143A+4F↑j
code:0000149B E5 3A                    mov     A, 0x3A
code:0000149D 64 03                    xrl     A, #3
code:0000149F 70 58                    jnz     code_14F9
code:00001A1 F5 76                     mov     0x76, A
```

- Convert the HEX to BIN
- 32k file for nRF24LU1
- Hunt for the encryption lib call
- Question, What is the key or where is the key
- Learn Intel 8051 Assembly

Why If It Really Breakable



- What if MouseJack team actually breaks the keyboard encryption
- Broken, will be broken forever
- It is possible to break the encryption. Why?

Before We Really End

Contact

xwings / tuya

Unwatch 2 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Wiki | Pulse | Graphs | Settings

No description or website provided. — Edit

52 commits | 1 branch | 0 releases | 1 contributor

Branch: master | New pull request | New file | Upload files | Find file | HTTPS | https://github.com/xwings | Download ZIP

xwings update logictech and elks code — Latest commit 06f3a61 4 days ago

archive	fixup	6 months ago
ctf	enjoy	2 months ago
mousejack	update logictech and elks code	4 days ago
README.md	Update README.md	6 months ago

README.md

all about reversing, exploit, ctf and misc

顾名思义 涂鸦

@mail: kj _ xandora _ net

- http://github.com/xwings/tuya
- Weibo: kaijern
- Twitter: kaijern

Questions