DRIVE IT LIKE YOU HACKED IT

DEFCON 23 [2015]
@SamyKamkar
http://samy.pl
Other Works

- Charlie Miller & Chris Valasek
- 2010: UCSD/UW Research (CD player, Bluetooth, etc)
- Relay Attacks (Amplification) on PKES
- Tesla talk later today!
- Cryptographic attacks on KeeLoq
- HiTag2 Immobilizer Disabling
- OpenGarages
- iamthecavalry
- Lots of others...
Thanks EFF!
GONE IN 60 SECONDS

www.wallpapers.cz
Part: NSCD-390GIV4
FCC ID: QVQ-QNRS283
Complies with FCC
Rules Part 15
Frequency: 315/390

Frequency: 315/390
use fcc.io, thanks Dominic Spill!
1 results were found that match the search criteria:
Grantee Code: qvq Product Code: -qnrs283

Displaying records 1 through 1 of 1.

<table>
<thead>
<tr>
<th>View Form</th>
<th>Display Grant</th>
<th>Display Correspondence</th>
<th>Applicant Name</th>
<th>Address</th>
<th>City</th>
<th>State</th>
<th>Country</th>
<th>Zip Code</th>
<th>FCC ID</th>
<th>Application Purpose</th>
<th>Final Action Date</th>
<th>Lower Frequency In MHz</th>
<th>Upper Frequency In MHz</th>
</tr>
</thead>
</table>

Perform Search Again

Please use the Submit Inquiry link at www.fcc.gov/labhelp to send any comments or suggestions for this site.

Federal Communications Commission
445 12th Street, SW
Washington, DC 20554
More FCC Contact Information...

Phone: 888-CALL-FCC (225-5322)
TTY: 888-TELL-FCC (835-5322)
Fax: 202-418-0232
E-mail: fccinfo@fcc.gov

- Privacy Policy
- Web Policies & Notices
- Customer Service Standards
- Freedom of Information Act
Operation Frequency: 390 MHz
Channel number: 1
Modulation type: ASK

Power Supply: DC 3V Supply

Applicant: Qinuo Electronics
Address: 3/F, Bldg. A, Y1 Fengze, Quanzhou

Manufacturer: Qinuo Electronics
Address: 3/F, Bldg. A, Y1 Fengze, Quanzhou
1 MHz - 6 GHz
half-duplex transceiver
raw I/Q samples
open source software / hardware
GNU Radio, SDR#, more
dope as shit

HackRF One
from Michael Ossmann
Replay Attack w/HackRF

- `hackrf_transfer -r 390_data.raw -f 3900000000` # listen
- `hackrf_transfer -t 390_data.raw -f 3900000000` # transmit
- # profit
- Don’t need baud rate
- Don’t need modulation/demodulation
- Can be within 20MHz
- Can act as a “raw” code grabber/replayer…but it’s more interesting than that.
RTL-SDR

24 - 1766 MHz
raw I/Q samples
RX only
RTL2832U
GNU Radio
(the stick shift of SDR)
waterfall views
demodulation
save to WAV

pretty
Linux & OS X Only

GQRX
Something happened
SDR# Works on Windows
Sorta kinda on OS X
rtl_fm, a simple narrow band FM demodulator for RTL2832 based DVB-T receivers

Use:  rtl_fm -f freq [-options] [filename]
      -f frequency_to_tune_to [Hz]
      use multiple -f for scanning (requires squelch)
      ranges supported, -f 118M:137M:25k
      [-M modulation (default: fm)]
      fm, wbmf, raw, am, usb, lsb
      wbmf = -M fm -s 170k -o 4 -A fast -r 32k -l 0 -E deemp
      raw mode outputs 2x16 bit IQ pairs
      [-s sample_rate (default: 24k)]
      [-d device_index (default: 0)]
      [-g tuner_gain (default: automatic)]
      [-l squelch_level (default: 0/off)]
      [-p ppm_error (default: 0)]
      [-E enable_option (default: none)]
      use multiple -E to enable multiple options
      edge:  enable lower edge tuning

terminal based
quick and easy
demodulates
### OET Exhibits List

8 Matches found for FCC ID QVQ-QNRS283

<table>
<thead>
<tr>
<th>View Attachment Exhibit Type</th>
<th>View Attachment Exhibit Type</th>
<th>Date Submitted to FCC</th>
<th>Display Type</th>
<th>Date Available</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cover letter</td>
<td>Cover Letter(s)</td>
<td>11/29/2014</td>
<td>pdf</td>
<td>11/29/2014</td>
</tr>
<tr>
<td>Cover letter</td>
<td>Cover Letter(s)</td>
<td>11/29/2014</td>
<td>pdf</td>
<td>11/29/2014</td>
</tr>
</tbody>
</table>

Please use the Submit Inquiry link at [www.fcc.gov/labhelp](http://www.fcc.gov/labhelp) to send any comments or suggestions for this site.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation Frequency</td>
<td>390 MHz</td>
</tr>
<tr>
<td>Channel number</td>
<td>1</td>
</tr>
<tr>
<td>Modulation type</td>
<td>ASK</td>
</tr>
<tr>
<td>Power Supply</td>
<td>DC 3V Supply</td>
</tr>
<tr>
<td>Applicant</td>
<td>Qinuo Electronics</td>
</tr>
<tr>
<td>Address</td>
<td>3/F, Bldg. A, Y1 Fengze, Quanzhou</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>Qinuo Electronics</td>
</tr>
<tr>
<td>Address</td>
<td>3/F, Bldg. A, Y1 Fengze, Quanzhou</td>
</tr>
</tbody>
</table>
Modulation Schemes
Modulation Schemes

2FSK

ASK (OOK)
ASK (OOK)
10-bit Garage
Fixed Code Garages

8 - 12 bit code

~2ms per bit + ~2ms delay

5 signals per transmission

\[((2 \times 12) \times 12) + ((2 \times 11) \times 11) + ((2 \times 10) \times 10) + ((2 \times 9) \times 9) + ((2 \times 8) \times 8)\] = 88576 bits

88576 bits * (2ms signal + 2ms delay) * 5 transmissions = 1771520ms = 1771secs = 29.5 minutes
1771 secs / 5 = 354.2 = 6 mins
354.2 secs / 2 = 177 secs = 3 mins

Thanks Mike Ryan!

Saturday, 3pm, Track Two
Hacking Electric Skateboards
Mike Ryan & Richo Healey
Where does one code end and the other begin?

Bit shift register?
Bit Shift Register

Code only clears one bit at a time while pulling in next bit

A 13 bit code tests two different 12 bit codes!

1000000000001
100000000000
1
De Bruijn Sequence

00110 (5 bits) tests all 4 different 2-bit sequences instead of 8 bits total

00110
00110
00110
00110

vs 00011110

Alphabet: \{0, 1\}
Subsequence length: 2

Subsequences:

\{0, 0\}  \{1, 0\}
-\{0, 1\}  \{1, 1\}

De Bruijn sequence:

\{0, 0, 1, 1\}
Brute forcing a 3-bit code

1 bit  10 bits  20 bits  30 bits  40 bits  50 bits

bit 0123456789012345678901234567890123456789001234567890

000---001---010---101---011---111---110---100--- <-- 48 bits

OpenSesame Attack

First, remove the wait periods (reduces 50%):  

1 bit  10 bits  20 bits  30 bits  40 bits  50 bits

bit 012345678901234567890123456789012345678901234567890

000001010101011111110100  <-- 24 bits

By overlapping (De Bruijn), we reduce another ~62%:

000 011
 001 111
 010 110
 101 100

0001011100  <-- 10 bits!
De Bruijn Sequence

For every 8 to 12 bit garage code

$((2^{12}) + 11) \times \frac{4ms}{2} = 8214ms = 8.214$ seconds
Yard Stick One
by Michael Ossmann
TI CC1111 chipset

rfcat
by atlas
Friday, 5pm, Track Two
Fun with Symboliks
Research Mode: enjoy the raw power of rflib currently your environment has an object called you interact with the rfcat dongle:

```python
>>> d.ping()
```

```python
>>> d.setFreq(4330000000)
```

```python
>>> d.setMdmModulation(MOD_ASK_OOK)
```

```python
>>> d.makePktFLEN(250)
```

```python
>>> d.RFxmit("HALLO")
```

```python
>>> d.RFrecv()
```

```python
>>> print d.reprRadioConfig()
```
At the local hacker space, Barbie has been working with Evelina where they've developed firmware for their favorite Mattel toy, the IM-ME, to perform RF jamming, automated sniffing, demodulation, and replay attacks on ISM bands under 1GHz.
Mattel IM-ME

TI CC1101 chipset
sub-GHz transceiver
screen, backlight, keyboard, stylish

Previously hacked by:
Dave
Michael Ossmann
Travis Goodspeed
Hacker Barbie
GoodFET
by Travis Goodspeed

open source JTAG adapter / universal serial bus interface
OpenSesame

based off of Michael Ossmann’s opensesame ASK transmitter
https://github.com/mossmann/im-me/tree/master/garage
Radica IM Me Wireless Handheld

1 viewed per hour

Item condition: New

Quantity: 1

Price: US $909.83

5 watching

Free delivery in 4 days

Hassle-free
Lessons

❖ Don’t use a ridiculously small key space (duh)
❖ Require a preamble/sync word for beginning of each key
❖ Use rolling codes…
Key Fob
Vehicle Commands

Vehicle Status
Diagnostics

Map
Location Services

Hands-Free Calling
Upgrade Now

Volt

Lock
Unlock
Success, 6:14 PM

Remote Start
Cancel Start

Horn & Lights
Stop Horn & Lights
RemoteLink Login
(base64 decoded)
SSL MITMA

- Raspberry Pi
- FONA GSM board
- mallory (SSL MITMA)
- dns spoofing (api.gm.com)
- iptables
- Alfa AWUS036h
- Edimax Wifi dongle
- pre-paid SIM card
802.11 Probe Requests
Lessons

❖ Validate certificates from CA

Press Release issued on 19 April 2010
Hongkong Post Certification Authority's root certificate included in Mozilla Firefox web browser

❖ Better yet, use certificate pinning and ignore CAs altogether

❖ Hash password with random salt on authentication (challenge-response)

❖ Always assume you’re on a hostile network
Key Fobs & Rolling Codes
National Semiconductor
“High Security Rolling Code” chip

Thanks Michael Ossmann for helping decipher this!
Rolling Codes

- PRNG in key and car
- Synced seed + counter
- Hit button, key sends code
- Hit button again, key sends next code
- If Eve replays the code, car rejects it because already used
- Should be difficult to predict
- Prevents replay attacks
Replaying Rolling Codes

- Capture signal while remote out of range from vehicle/garage
- Replay later
- This is lame since we have to have access to the key, and it has to be far from the car
We’re Jammin
Jam + Listen, Replay

- Jam at slightly deviated frequency
- Receive at frequency with tight receive filter bandwidth to evade jamming
- User presses key but car can’t read signal due to jamming
- Once we have code, we stop jamming and can replay
- But…once user does get a keypress in, new code invalidates our code!
Jam+Listen(1), Jam+Listen(2), Replay (1)

- Jam at slightly deviated frequency
- Receive at frequency with tight receive filter bandwidth to evade jamming
- User presses key but car can’t read signal due to jamming
- User presses key again — you now have two rolling codes
- Replay first code so user gets into car, we still have second code
<table>
<thead>
<tr>
<th>0/11 bits</th>
<th>0/8 bits</th>
<th>0/20/24 bits</th>
<th>4 bits</th>
<th>24/36 bits</th>
<th>0/8 bits</th>
<th>1 bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preamble</td>
<td>Sync</td>
<td>Key ID</td>
<td>Data</td>
<td>Dynamic</td>
<td>Parity</td>
<td>Stop</td>
</tr>
<tr>
<td></td>
<td>Field</td>
<td>Field</td>
<td>Field</td>
<td>Code</td>
<td>Field</td>
<td>Bit</td>
</tr>
</tbody>
</table>

**FIGURE 4. Normal Data Frame Configuration**

The primary use of the data field is to indicate which key switch has been pressed. Since each key switch input can be associated with a particular application, the decoder can determine which function to initiate.

**DYNAMIC CODE FIELD**

The dynamic code field is transmitted with every frame, and its length is programmable. If DynSize = 0, a 24-bit field is sent; if DynSize = 1, a 36-bit field is sent. Its function is to provide a secure dynamic code which changes with each new transmission. The field is the result of combining the

**Protocol Abuse**
Teensy 3.1

CC1101

RollJam

(I’m bad at names)
National Semiconductor
“High Security Rolling Code” chip

Thanks Michael Ossmann for helping decipher this!
Lessons

- Encrypt/hash the button/action
- HMAC to prevent bit flipping if encrypted
- Use time-based algorithm (e.g. RSA SecurID [20 years old], “Dual KeeLoq” does this as of 2014)
- OR challenge/response via transceivers instead of one-way communication
- Many vehicles have keys that RX+TX yet the remote unlock signal is still one-way and not timing based
Thank You!!!

YOU!
EFF
Michael Ossmann
Travis Goodspeed
Andy Greenberg
atlas of d00m
My mom
Defcon
TI
#hackrf
#ubertooth
Charlie Miller
Chris Valasek
Mike Ryan
Andrew Crocker
Nate Cardozo
Kurt Opsahl

@SamyKamkar
http://samymyl
http://samymyl/youtube