Boston Infinite Money Glitch:

Hacking Transit Cards Without Ending Up In Handcuffs

Matthew Harris, Zachary Bertocchi, Scott Campbell, Noah Gibson

Boston Infinite Money Glitch:

Hacking Transit Cards Without Ending Up In Handcuffs

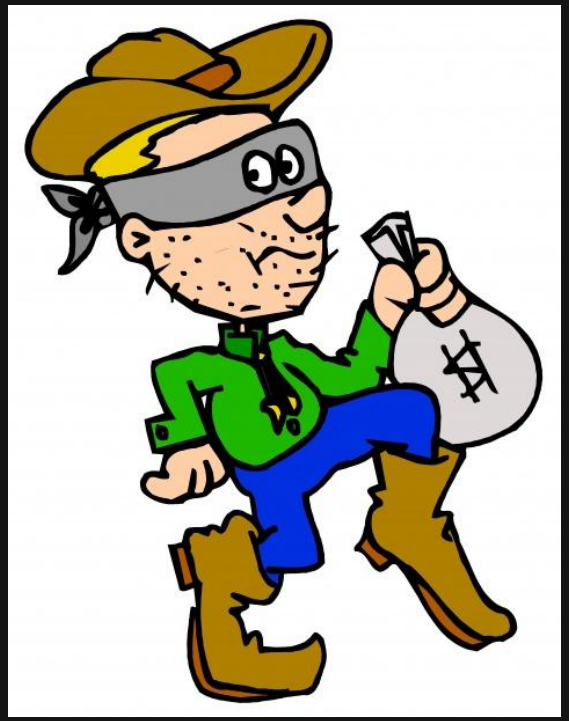
Matthew Harris, Zachary Bertocchi, Scott Campbell, Noah Gibson

* The MBTA's lawyers have kindly asked us to say the following:

"The MBTA does not endorse or encourage this type of conduct. evading fares and/or hacking the MBTA fare system is illegal and the MBTA takes these matters seriously. Anyone caught engaging in this type of behavior will be referred to the appropriate law enforcement agency. The MBTA has implemented mitigation measures to address some of these concerns."

whoarewe

- Vocational high schoolers from Medford, MA
- ~~Professional~~ Amateur miscreants
- People who don't like being told what to do
- Underwater robotics enjoyers



Matthew Harris

Zack Bertocchi

Scott Campbell

Noah Gibson

“What am I getting out of this?”

- Free rides on the MBTA
- A lesson on how to reverse engineer a transit card
- A lesson on how to not end up in handcuffs for the above two items
- A fun story, hopefully

THIS IS ILLEGAL

(don’t do it)

(or at least don’t tell us about it)

Introduction

Meet the T

- Massachusetts Bay Transportation Authority - The public transit agency in Boston
- First transit system in the country

Image Credit: MBTA

Image Credit: MBTA

Sounds great! How can I hop on?

Option 1: Traditional fare evasion

Sounds great! How can I hop on?

Option 1: Traditional fare evasion

- Hop the gates

Sounds great! How can I hop on?

Option 1: Traditional fare evasion

- Hop the gates
- Force the gates open

Sounds great! How can I hop on?

Option 1: Traditional fare evasion

- Hop the gates
- Force the gates open
- Too hard to explain, just look

Sounds great! How can I hop on?

Option 1: Traditional fare evasion

- Hop the gates
- Force the gates open
- Too hard to explain, just look

This works too, I guess

Option 2: Pay

- Subway fare - $2.40
- Bus fare - $1.70

The MBTA has authorized us to use the CharlieCard and CharlieTicket logos

The Problem With These Strategies

Traditional Fare Evasion

- Too easy
- Boring
- Might get caught

Paying

- Too legal
- Way too boring
- Costs money

What to do?

Hack the fare system

Hack the fare system

duh

Part 1: Humble Beginnings

CharlieCard

From Wikipedia, the free encyclopedia

The **CharlieCard** is a contactless smart card used for fare payment for transportation in the Boston area. It is the primary payment method for the Massachusetts Bay Transportation Authority (MBTA) and several regional public transport systems in the U.S. state of Massachusetts.

The card was introduced on December 4, 2006, to enhance the technology of the transit system and eliminate the burden of carrying and collecting tokens.[3] It replaces the metal token, the last one of which was sold at Government Center station on December 6, 2006.[4] It is named after a fictional character in the folk song "M.T.A.", often called "Charlie on the MTA", which concerns a man forever trapped on the Boston subway system – then known as the Metropolitan Transit Authority (MTA) – because he cannot pay the 5-cent surcharge required to leave the train.

In 2022, the original CharlieCard system will be replaced during the "Automated Fare Collection 2.0" project, a system similar to the London Oyster Card. The new system will allow payments with contactless cards and smartphones, as well as new CharlieCards.[5]

Contents [hide]

History [edit]

The CharlieCard is named after the lead character in the 1948 protest folk music song, "M.T.A.". The song was written to protest a fare increase in the form of an extra five cent exit fare for longer rides and was later made popular by the Kingston Trio in 1959.[6][7] One of the rejected names for the farecard system was "The Fare Cod", a pun on both the way locals might pronounce "Card" and the fish that was once integral to the Massachusetts economy, and also a reference to other transit cards named for ocean animals, such as London's Oyster and Hong Kong's Octopus. Another rejected name was T Go card with the T being the symbol for the MBTA.[8]

CharlieCards work on the MBTA's subway and bus services, most of which were converted in 2006. Token sales ended on December 6, 2006.[9] The final fare-controlled station to be converted was Fields Corner station on December 22, 2006.[10] They were originally expected to be usable on MBTA commuter rail and ferry boat services by December 2008,[11] with testing on the Commuter Rail originally planned for summer 2008.[12] However,

CharlieCard

Location	Boston, Massachusetts, U.S.
Launched	December 4, 2006
Technology	Contactless smart card MIFARE
Manager	Massachusetts Bay Transportation Authority
Currency	USD
Validity	Massachusetts Bay Transportation Authority Berkshire Regional Transit Authority Brockton Area Transit Authority Cape Ann Transportation Authority Cape Cod Regional Transit Authority Lowell Regional Transit Authority Merrimack Valley Regional Transit Authority MetroWest Regional Transit Authority (Discontinued support for CharlieCard since May 2, 2022, due to new fare system. New primary method for payment on MWRTA is named "Catch Card" [1].) Montachusett Regional Transit Authority (Discontinued support for CharlieCard since June 1, 2022, due to new fare system. New primary method for payment

August 2021

Security concerns [edit]

See also: Massachusetts Bay Transportation Authority v. Anderson

Security flaws in the CharlieCard technology were studied and reported in a presentation by Henryk Plötz and Karsten Nohl at the Chaos Communication Congress in December 2007, which described a partial reverse-engineering of the algorithm used in the MIFARE Classic chip.[71] The MIFARE Classic smartcard[72] from NXP Semiconductors, owned by Philips, was reported as compromised in March 2008 by a group of researchers led by Karsten Nohl, a Ph.D. student in the Department of Computer Science, University of Virginia.[73][74][75]

In addition, the security used on the mag-stripe CharlieTickets was broken by a team of MIT students. They were scheduled to give a talk about their findings at DEFCON 16 in August 2008,[76] but were stopped after a federal lawsuit was filed against them by the MBTA, which resulted in a restraining order being issued.[77][78] However, their presentation had already been published by DEFCON before the complaint was filed.[79] On August 19, the court ruled the students could give their presentation.[80]

Other MIT students leveraged the technology behind Charlie Cards in 2013, with the development of Sesame Ring, a wearable ring embedded with an RFID tag that would save riders time in passing through MBTA station faregates.[81] The students formed a company called Ring Theory and funded development of the product using a Kickstarter campaign. The Sesame Ring can be ordered online, or purchased in the MBTA Gift Store in Cambridge.[82] The product was developed with full cooperation from the MBTA.[83]

In 2022, it was revealed that the NFC chip in some Android smartphones could interact with CharlieCards, including duplicating data from one card to a blank card. The MBTA indicated that its software systems detected a small number of such duplicated cards – about ten per month – which were then deactivated.[84]

The OGs

- 15 years ago
- Some MIT students found a vulnerability in the CharlieCards and Tickets
- Figured out how to clone and forge CharlieTickets
- Wanted to present at DEF CON
- How’d they get thanked for their research?

UNITED STATES DISTRICT COURT
DISTRICT OF MASSACHUSETTS

MASSACHUSETTS BAY TRANSPORTATION AUTHORITY

Plaintiff

v.

ZACK ANDERSON, RJ RYAN, ALESSANDRO CHIESA, and the MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Defendants

Civil Action No. No. 08- 11364-GAO

DECLARATION OF SCOTT HENDERSON

1. I am the Systems Project Manager for the Automated Fare Collection System for the plaintiff, Massachusetts Bay Transportation Authority ("MBTA").

2. My duties include (a) responsibility for the computer systems resident in all of the devices within the Automated Fare Collection ("AFC") System, and (b) responsibility for all software application systems within the AFC System. I have included further information concerning my qualifications, relevant to this Declaration, in paragraphs 28 through 37.

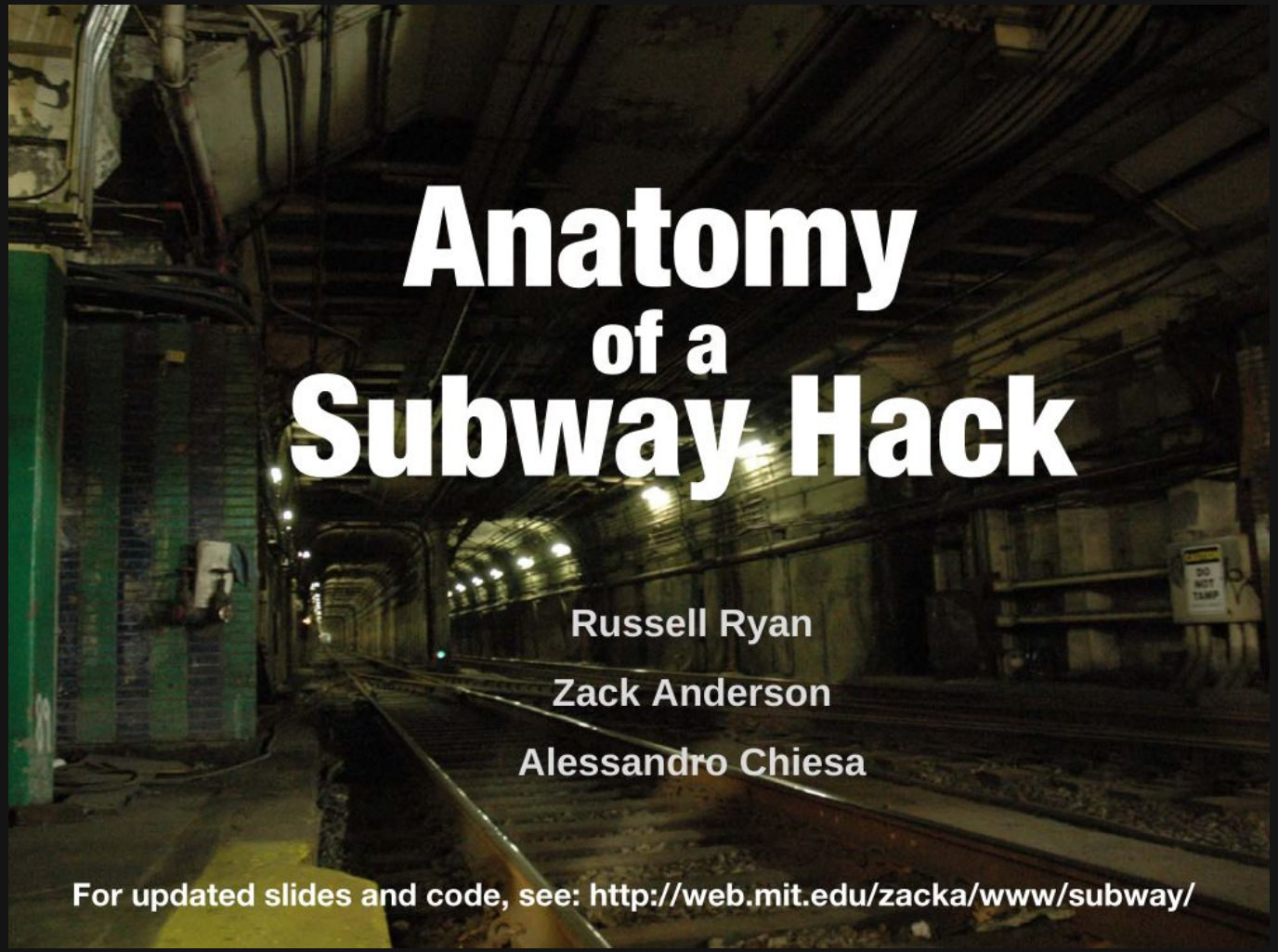
3. I make this declaration based on my personal knowledge and a review of MBTA business records concerning the matters set out below.

Requests To The MIT Undergrads For Their DEFCON Presentation Materials

4. I have reviewed the documents referred to in the Complaint as the " Initial Announcement" and the "Revised Announcement". I use the term "MIT Undergrads" to refer to the defendants, Zack Anderson, RJ Ryan, and Alessandro Chiesa.

DefCon: Boston Subway Officials Sue to Stop Talk on Fare Card Hacks — Update: Restraining Order Issued; Talk Cancelled

BY KIM ZETTER 08.08.08 | 11:45 PM | PERMALINK


Anatomy
of a
Subway Hack
Russell Ryan
Zack Anderson
Alessandro Chiesa
For updated slides and code, see: http://web.mit.edu/zacka/www/subway/

Let’s try to be like them

Maybe minus the lawsuit part

Step 1: Go for the low hanging fruit

- CharlieTickets have awful security
- MIT students found how to clone and reverse engineer them
- Seems easy enough
- Free rides 4 life!

Step 2: Choose your weapon

- MSR206
- 88 bucks
- A bit pricey, but it’s an investment
- Should we go ahead and buy it?

Maybe not

“We’re phasing out the old CharlieTickets for new, tappable versions. Riders can purchase the new CharlieTickets at new fare vending machines across the system.”

September 2021

Time for a new plan

Meet the CharlieCard

- Contactless smart card
- Introduced in 2004
- Add money at vending machines
- Tap cards at fare gates

But how does it work?

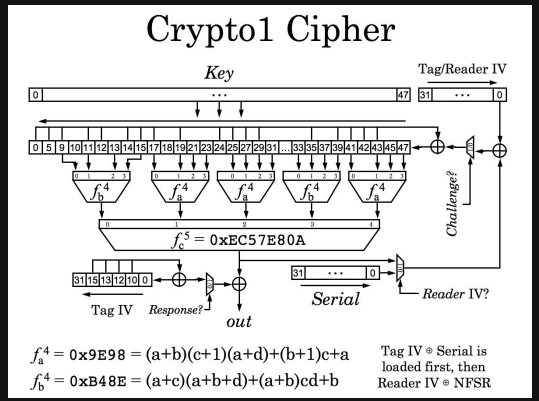
- Not 100% sure yet
- We know it uses NFC
- Probably uses MIFARE under the hood

MIFARE?

- Standard for data storage and communication
- Made by NXP
- Lots of different flavors of MIFARE
- Most ~~in~~famous flavor: MIFARE Classic

MIFARE Classic

- Oldest flavor
- Crypto-1 encryption algorithm: Proprietary, 48-bit, secured through obscurity
- History of lawsuits around it



How it's supposed to work

How it actually works

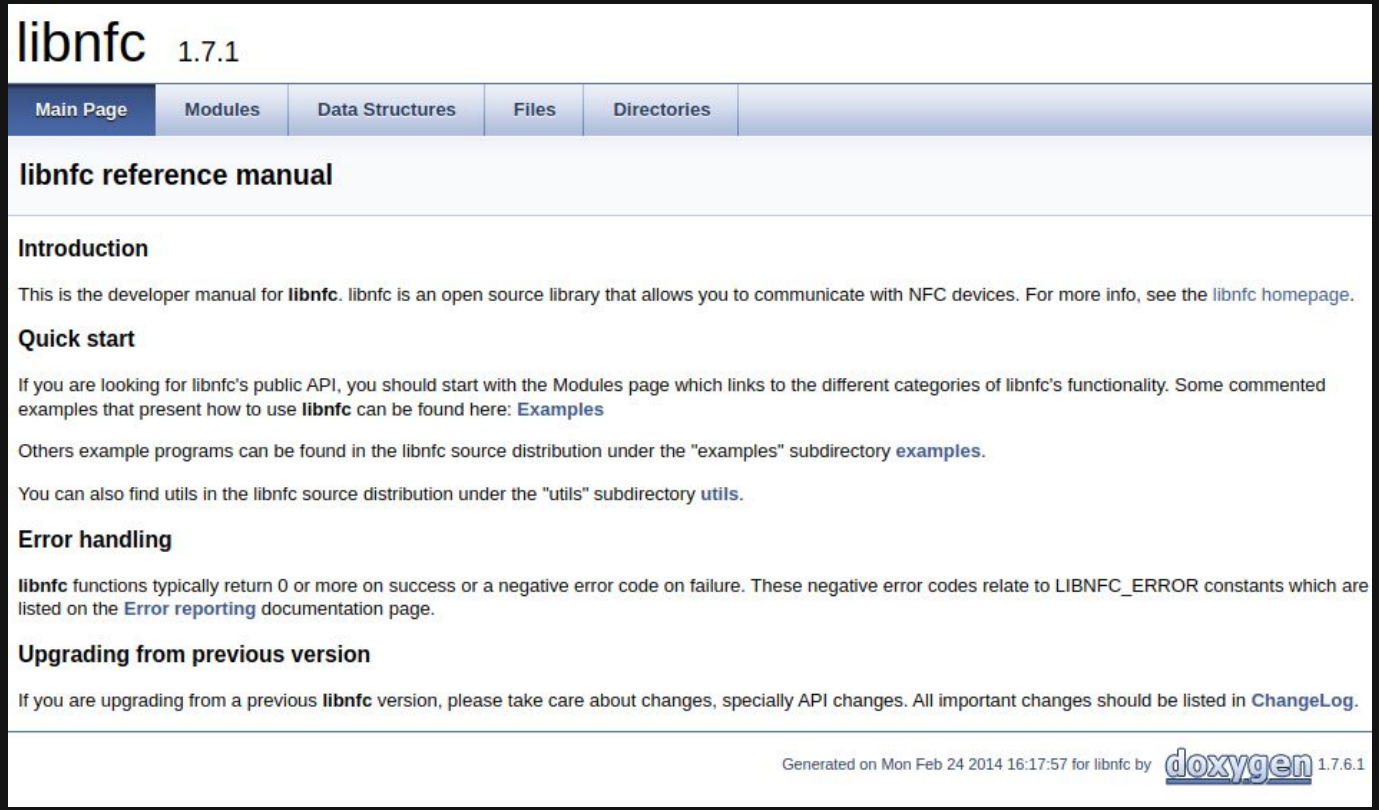
So how do we talk to the cards?

Step 1: Grab an NFC reader

- ACR122U
- $40
- USB
- Software exists

September 2021

Step 2: Figure out how to use it



libnfc 1.7.1

Main Page | Modules | Data Structures | Files | Directories

libnfc reference manual

Introduction

This is the developer manual for **libnfc**. libnfc is an open source library that allows you to communicate with NFC devices. For more info, see the libnfc homepage.

Quick start

If you are looking for libnfc's public API, you should start with the Modules page which links to the different categories of libnfc's functionality. Some commented examples that present how to use **libnfc** can be found here: **Examples**

Others example programs can be found in the libnfc source distribution under the "examples" subdirectory **examples**.

You can also find utils in the libnfc source distribution under the "utils" subdirectory **utils**.

Error handling

libnfc functions typically return 0 or more on success or a negative error code on failure. These negative error codes relate to LIBNFC_ERROR constants which are listed on the **Error reporting** documentation page.

Upgrading from previous version

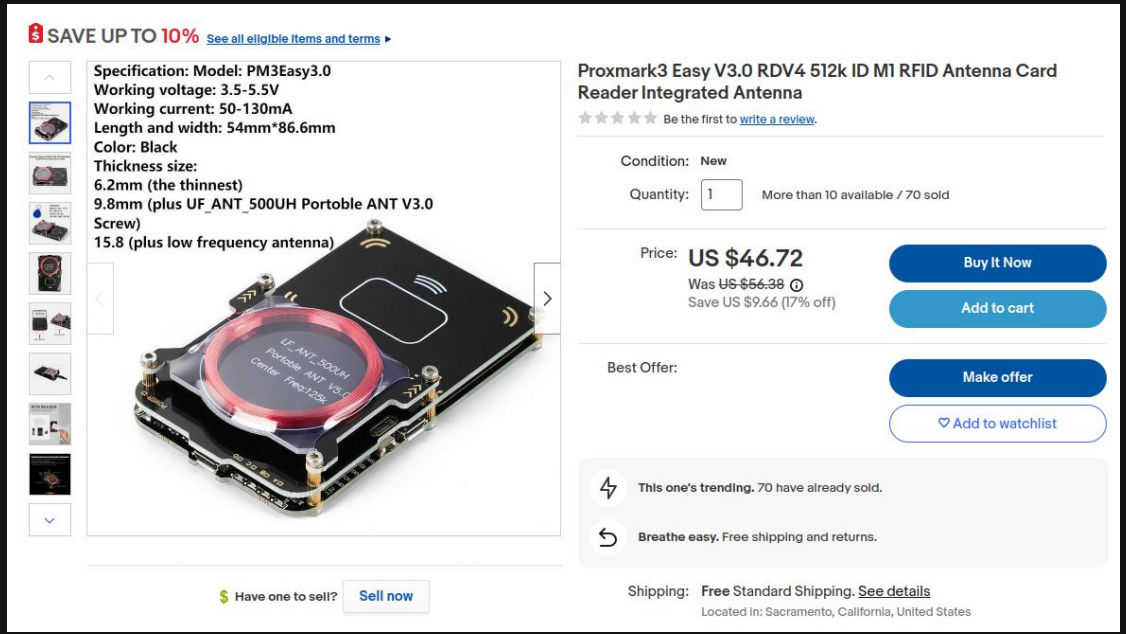
If you are upgrading from a previous **libnfc** version, please take care about changes, specially API changes. All important changes should be listed in **ChangeLog**.

Generated on Mon Feb 24 2014 16:17:57 for libnfc by doxygen 1.7.6.1

Step 3: Send it back cause you don’t know how to use it

Step 4: Buy another reader

- Proxmark3
- $40-80
- Truly top-notch software
- Decently easy to use



October 2021

Wait a minute, it doesn’t work!

Step 5: Buy another Proxmark

Proxmark3 Easy RFID Copier NFC RFID Detector Duplicator Reader Writer Cloner 3.0 512K Memory with 2 USB IC ID 125 kHz 13.56 Mhz

★★★★★ 5.0 2 Reviews 10 Sold

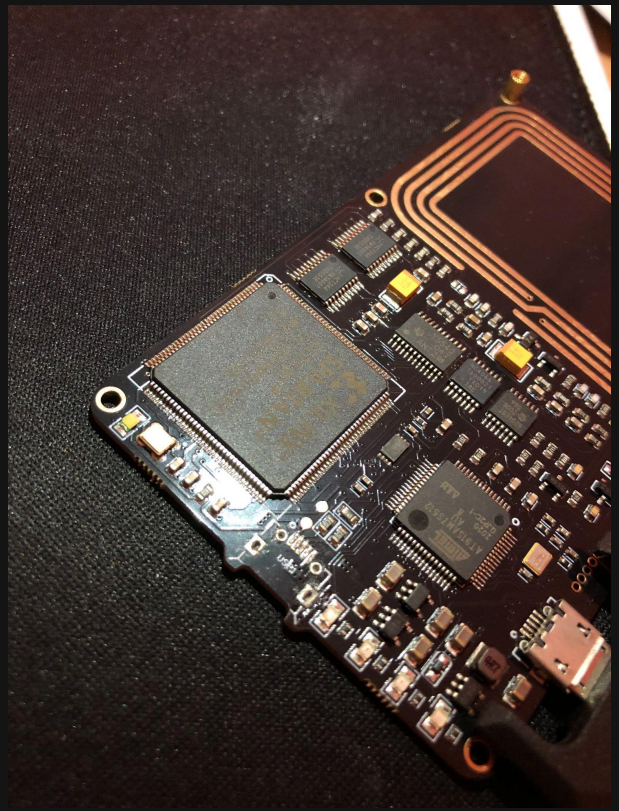
$39.75 ~~$46.76~~ 15% off

2% Off
Store Discount

$1.00 Off
Store Coupon

It works!!!

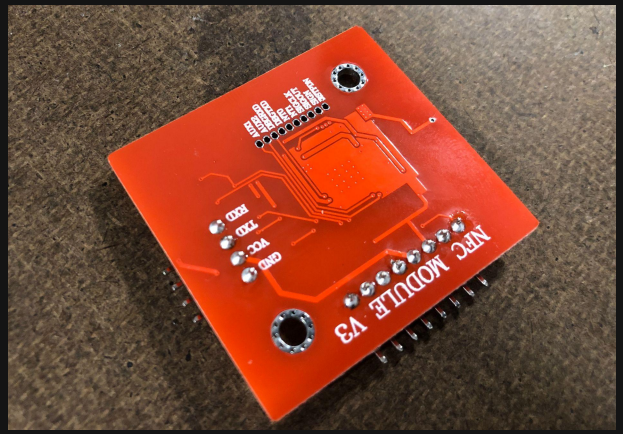
fuck



November 2021

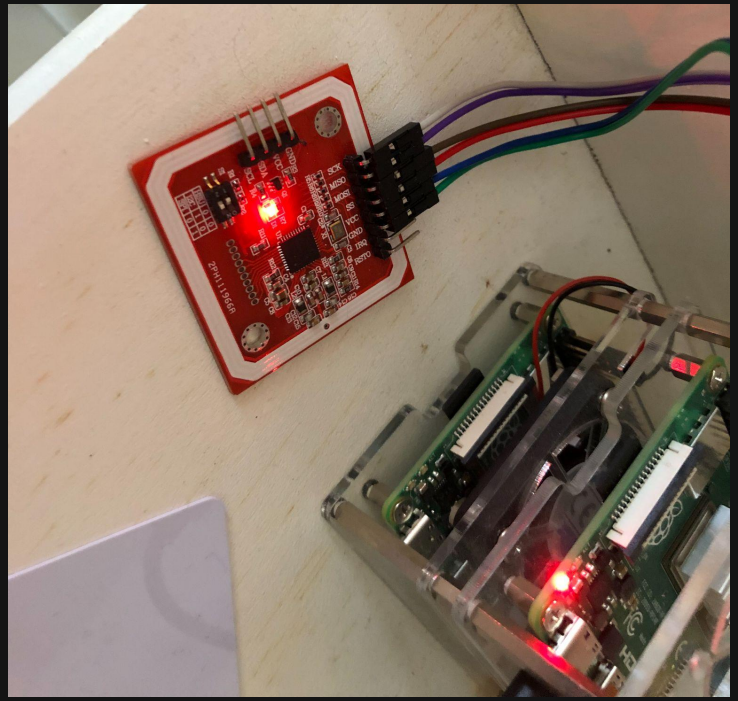
Step ¯_(ツ)_/¯: Buy another reader

- PN532
- $9
- Hope you like soldering
- Only works with Raspberry Pi



November 2021

It works! For real this time



Next Step: Get the keys

- Bad encryption is still encryption
- Still need keys to read and write
- How do we get them?

Looks promising

Card-only Attacks

- Nested Attack
  - Introduced in 2009 by Nijmegan Oakland and Implemented by Nethemba with the MFOC tool.

- Dark-Side Attack
  - Introduced in 2009 by Nicolas Courtois and implemented by Andrei Costin with the MFCUK.

Let’s give them a try!

mfoc

- Implements the nested attack
- Does some fancy magic to find a card’s keys
- Let’s try it!

mfoc

- Implements the nested attack
- Does some fancy magic to find a card’s keys
- Let’s try it!

mfoc

- Implements the nested attack
- Does some fancy magic to find a card's keys
- Let's try it!
- Requires an initial key

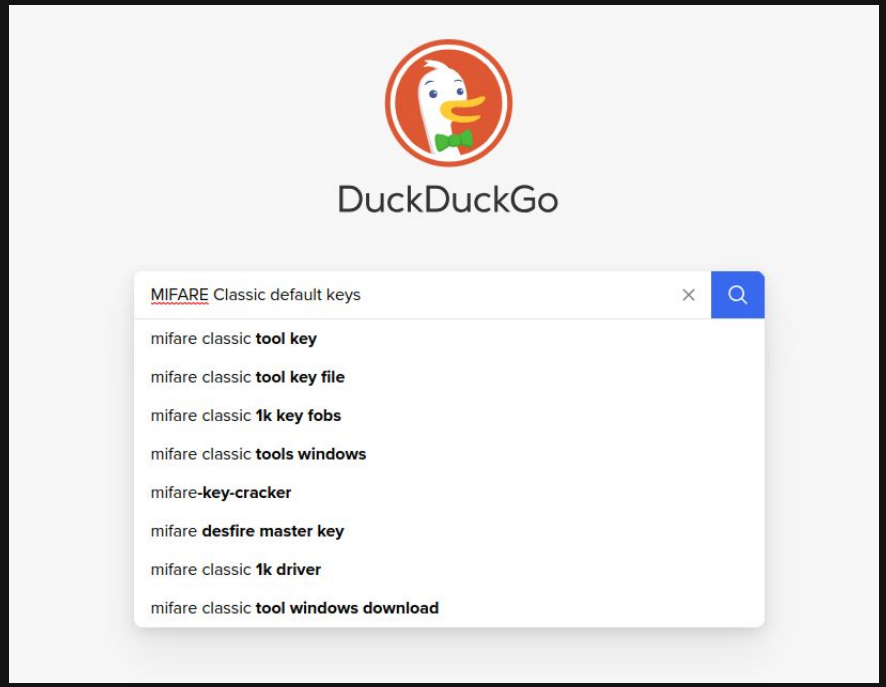
```
Try to authenticate to all sectors with default keys...
Symbols: '.' no key found, '/' A key found, '\' B key found, 'x' both keys found
[Key: ffffffffffff] -> [................]
[Key: a0a1a2a3a4a5] -> [................]
[Key: d3f7d3f7d3f7] -> [................]
[Key: 000000000000] -> [................]
[Key: b0b1b2b3b4b5] -> [................]
[Key: 4d3a99c351dd] -> [................]
[Key: 1a982c7e459a] -> [................]
[Key: aabbccddeeff] -> [................]
[Key: 714c5c886e97] -> [................]
[Key: 587ee5f9350f] -> [................]
[Key: a0478cc39091] -> [................]
[Key: 533cb6c723f6] -> [................]
[Key: 8fd0a4f256e9] -> [................]

Sector 00 - Unknown Key A               Unknown Key B
Sector 01 - Unknown Key A               Unknown Key B
Sector 02 - Unknown Key A               Unknown Key B
Sector 03 - Unknown Key A               Unknown Key B
Sector 04 - Unknown Key A               Unknown Key B
Sector 05 - Unknown Key A               Unknown Key B
Sector 06 - Unknown Key A               Unknown Key B
Sector 07 - Unknown Key A               Unknown Key B
Sector 08 - Unknown Key A               Unknown Key B
Sector 09 - Unknown Key A               Unknown Key B
Sector 10 - Unknown Key A               Unknown Key B
Sector 11 - Unknown Key A               Unknown Key B
Sector 12 - Unknown Key A               Unknown Key B
Sector 13 - Unknown Key A               Unknown Key B
Sector 14 - Unknown Key A               Unknown Key B
Sector 15 - Unknown Key A               Unknown Key B
mfoc: ERROR:

No sector encrypted with the default key has been found, exiting..
```

mfcuk

- Implements the dark side attack
- Finds the keys from out of nowhere
- Let’s try it!

```
mfcuk - 0.3.8
Mifare Classic DarkSide Key Recovery Tool - 0.3
by Andrei Costin, zveriu@gmail.com, http://andreicostin.com

WARN: cannot open template file './data/tmpls_fingerprints/mfcuk_tmpl_skgt.mfd'
WARN: cannot open template file './data/tmpls_fingerprints/mfcuk_tmpl_ratb.mfd'
WARN: cannot open template file './data/tmpls_fingerprints/mfcuk_tmpl_oyster.mfd'

INFO: Connected to NFC reader: ACS / ACR122U PICC Interface


VERIFY:
       Key A sectors: 0 1 2 3 4 5 6 7 8 9 a b c d e f
       Key B sectors: 0 1 2 3 4 5 6 7 8 9 a b c d e f

RECOVER:  0
```

Last ditch effort



2177 lines (2176 sloc) | 29.1 KB

```
1   #
2   # Mifare Default Keys
3   #   -- iceman fork version --
4   #   -- contribute to this list, sharing is caring --
5   #
6   # Default key
7   FFFFFFFFFFFF
8   #
9   # Blank key
10  000000000000
11  #
12  # NFC Forum MADkey
13  A0A1A2A3A4A5
14  #
```

holy shit

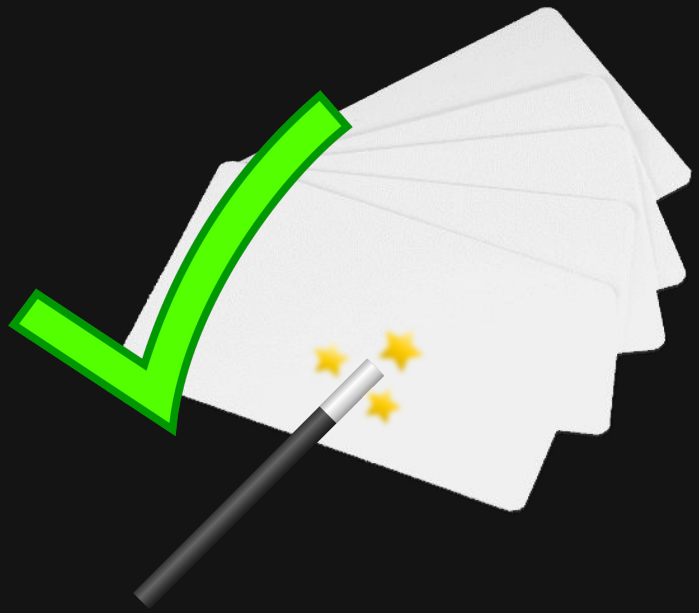
```
Sector 00 - Found   Key A: 3060206f5b0a Found   Key B: f1b9f5669cc8
Sector 01 - Found   Key A: 5ec39b022f2b Found   Key B: f662248e7e89
Sector 02 - Found   Key A: 5ec39b022f2b Found   Key B: f662248e7e89
Sector 03 - Found   Key A: 5ec39b022f2b Found   Key B: f662248e7e89
Sector 04 - Found   Key A: 5ec39b022f2b Found   Key B: f662248e7e89
Sector 05 - Found   Key A: 5ec39b022f2b Found   Key B: f66224ee1e89
Sector 06 - Found   Key A: 5ec39b022f2b Found   Key B: f66224ee1e89
Sector 07 - Found   Key A: 5ec39b022f2b Found   Key B: f66224ee1e89
Sector 08 - Found   Key A: 3a09594c8587 Found   Key B: 62387b8d250d
Sector 09 - Found   Key A: f238d78ff48f Found   Key B: 9dc282d46217
Sector 10 - Found   Key A: afd0ba94d624 Found   Key B: 92ee4dc87191
Sector 11 - Found   Key A: b35a0e4acc09 Found   Key B: 756ef55e2507
Sector 12 - Found   Key A: 447ab7fd5a6b Found   Key B: 932b9cb730ef
Sector 13 - Found   Key A: 1f1a0a111b5b Found   Key B: ad9e0a1ca2f7
Sector 14 - Found   Key A: d58023ba2bdc Found   Key B: 62ced42a6d87
Sector 15 - Found   Key A: 2548a443df28 Found   Key B: 2ed3b15e7c0f

We have all sectors encrypted with the default keys..
```

Let’s grab a binary dump

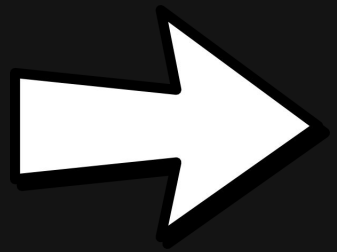
Part 2: Cloning Cards

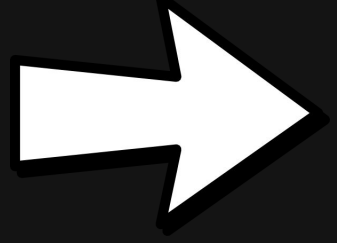
November 2021

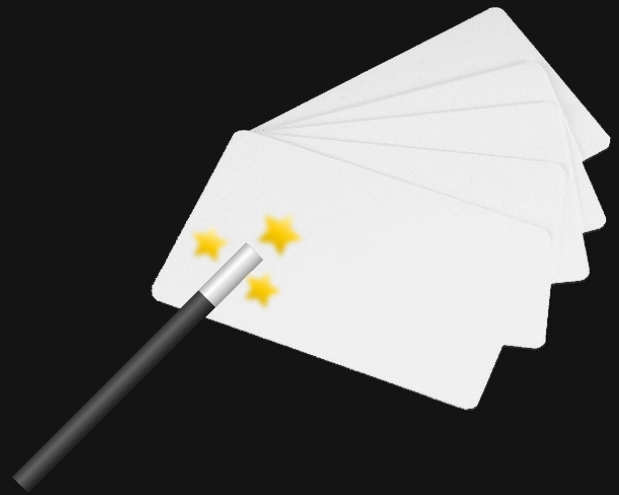
Magic? I love magic!!!




CharlieCard
Massachusetts Bay Transportation Authority
5 -243793061
G0016127454
Exp. 12/25








CharlieCard
Transit Value - Adult
Remaining Amount: $ 5.40
Please Make Your Selection
Buy Transit Value
1 or 7 Day LinkPass
Calendar Based Monthly Passes
Quick Ticket
Card / Ticket Information
Go Back
Cancel
START AUDIO PRESS 5

Wait wait… where’s the source of truth here?

The card

Let's automate it!

nvim main.c

main.c

```c
}

static void print_usage() {
    printf("Usage: ");
    printf("charlie-clone [options] <file>\n");
    printf("  -h, --help    Show this help page\n");
    printf("  -d, --dump    Save the contents of the card to a file\n");
    printf("  -f, --file    Read the contents of a file instead of a card,"
           "and write them to a blank card\n");

    printf("Examples: \n");
    printf("  charlie-clone -f card_data.mfd    Read the data from the file");
    printf("card_data.mfd and write it to a blank card\n");
    printf("  charlie-clone -d card_data.mfd    Save the contents of the card");
    printf("to the file card_data.mfd\n");
}

int main(int argc, char **argv) {
    int option;
    int option_index;
    bool dumping_to_file = false;
    char *dump_file_name = "";
    bool reading_from_file = false;
    char *read_file_name = "";

    static struct option long_options[] = {{"dump", required_argument, 0, 'd'},
                                           {"file", required_argument, 0, 'f'},
                                           {"help", no_argument, 0, 'h'},
                                           {0, 0, 0, 0}};
    if (argc > 3) {
        printf("Error: Too many arguments!\n");
        print_usage();
        exit(1);
    }

    while (true) {
        option = getopt_long(argc, argv, "hd:f:", long_options, &option_index);
        if (option == -1) {
            break;
        }
        switch (option) {
        case 'h':
            print_usage();
            exit(0);
            break;
        case 'd':
            dumping_to_file = true;
            dump_file_name = optarg;
            break;
        case 'f':
            reading_from_file = true;
            read_file_name = optarg;
            break;
```

NORMAL main.c c utf-8 67% ln :377/556 :79 W:12W:4(L23)

```
~/Projects/charlie-clone/src charlie-clone
charlie-clone v1.0.0

NFC reader: ACS / ACR122U PICC Interface opened
Place the CharlieCard on the reader!
Card found!
Found CharlieCard:
ISO/IEC 14443A (106 kbps) target:
    ATQA (SENS_RES): 00  04
       UID (NFCID1): f4  22  89  34
      SAK (SEL_RES): 08

[############################################################] 100%

Please remove card from reader
Card removed

Please place a blank magic card on the reader
Card found!

Found magic card:
ATQA (SENS_RES): 00  04
   UID (NFCID1): f4  22  89  34
  SAK (SEL_RES): 88

Card unlocked!
[############################################################] 100%
~/Projects/charlie-clone/src
```

November 2021

Infinite Money??? Not so fast…

- Requires upfront investment
- Might set off some fraud alarms
- When your card gets disabled you need to PAY to buy another one

Cloning Cards is Cool But Y'know What's Cooler?

Forging cards

Part 3: Cracking the Code

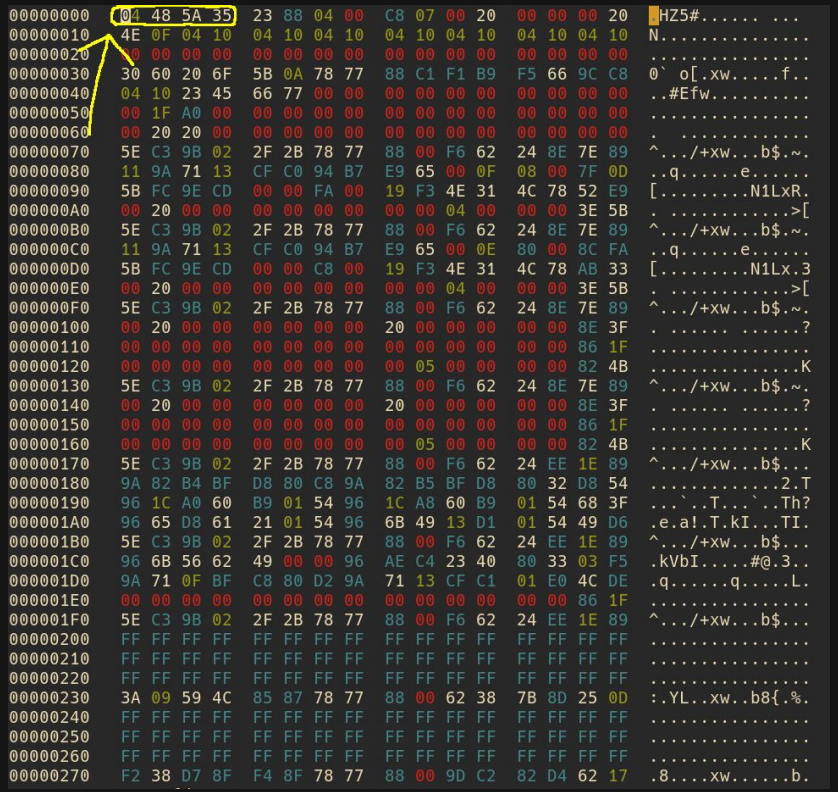
November 2021

Matty

Reverse Engineering a CharlieCard: Where do we start?

Let’s take a look at the hex dump!

Looks like the serial number

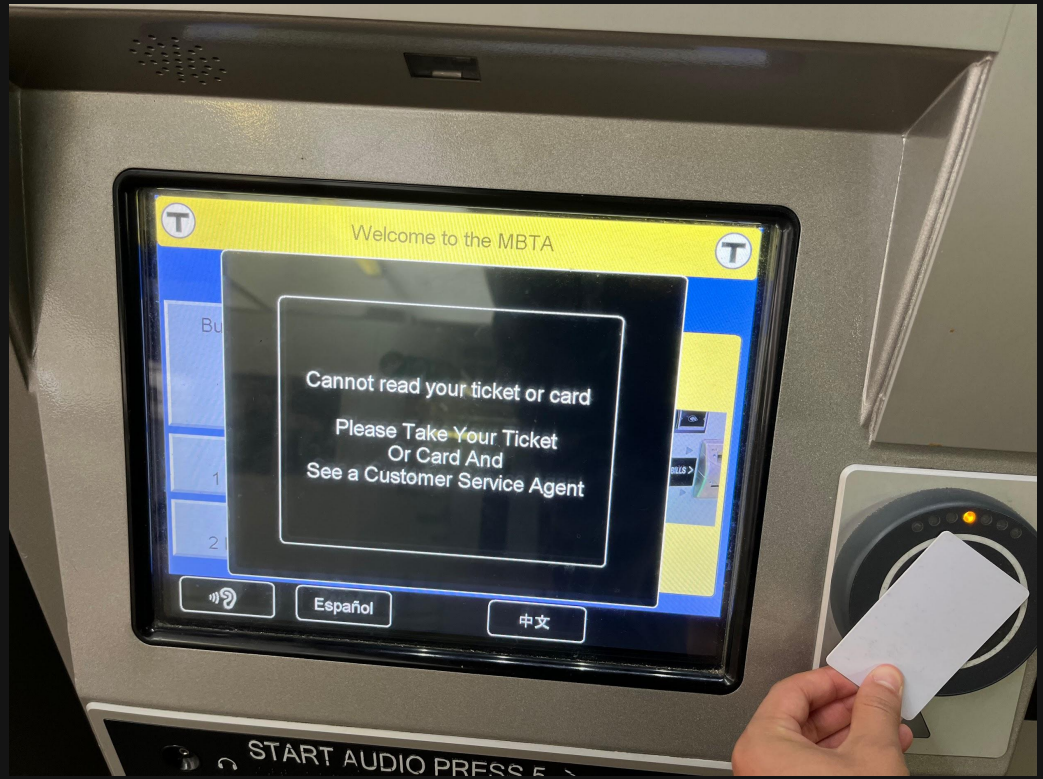


Let’s try changing some data

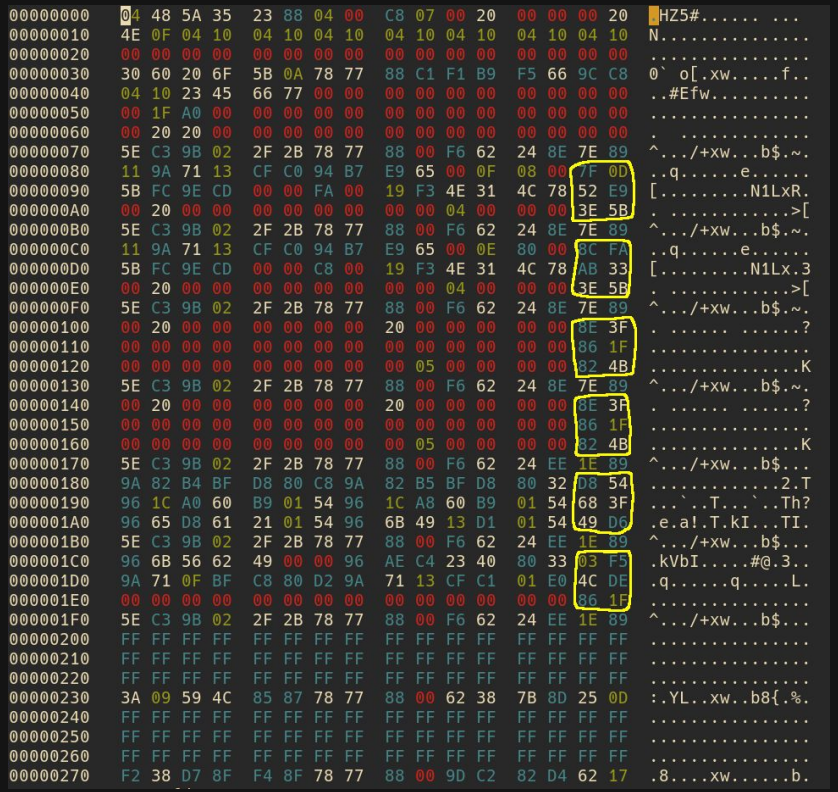
```
00 20 00 00  00 00 00 00  20 00 00 00  00 00 8E 3F
00 00 00 00  00 00 00 00  00 00 00 00  00 00 86 1F
00 00 00 00  00 00 00 00  00 05 00 00  00 00 82 4B
```

```
00 20 00 00  00 00 00 00  20 00 00 00  00 00 8E 3F
13 37 DE AD  BE EF 00 00  00 00 00 00  00 00 86 1F
00 00 00 00  00 00 00 00  00 05 00 00  00 00 82 4B
```

hrm


Welcome to the MBTA
Cannot read your ticket or card
Please Take Your Ticket
Or Card And
See a Customer Service Agent
Español
中文
START AUDIO PRESS 5

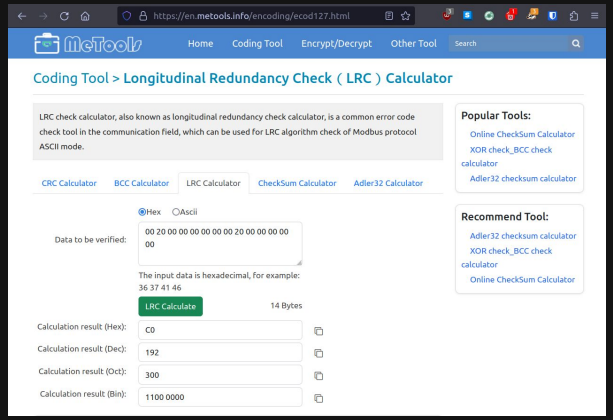
Smells like a checksum

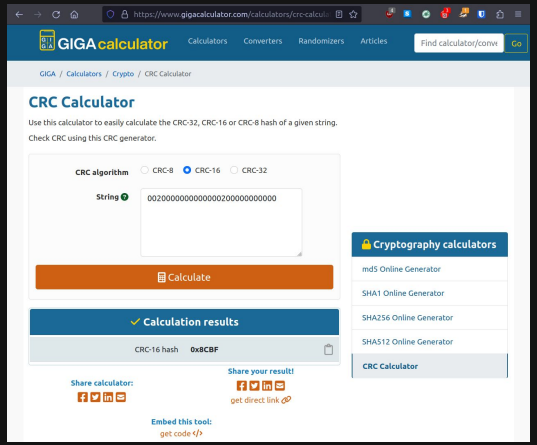


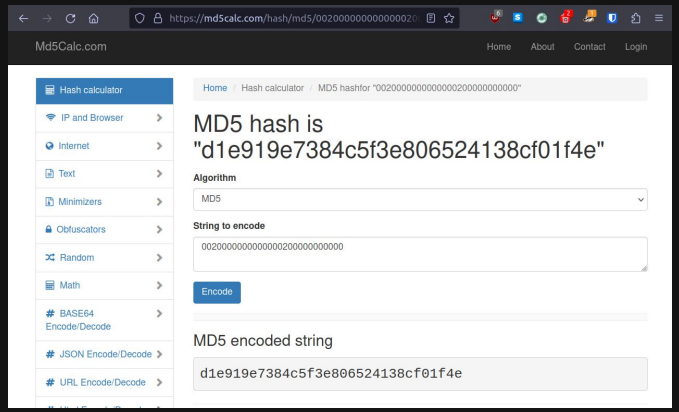
A checksum? Can’t be too hard to crack, right?

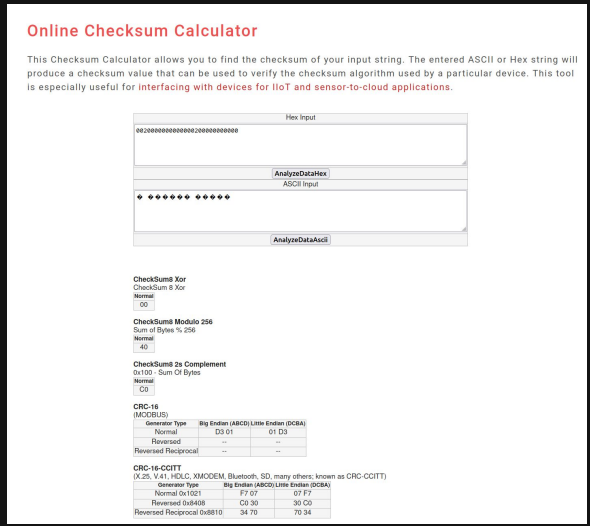
- Maybe it's a standard checksum
- Why wouldn’t they use a standard, secure algorithm?
- Let’s try some common algorithms!

Some common algorithms

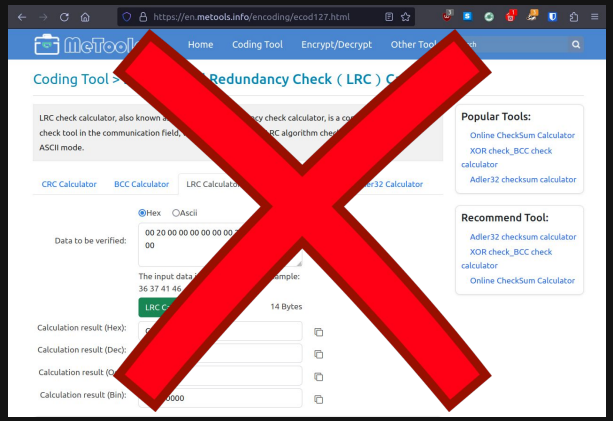


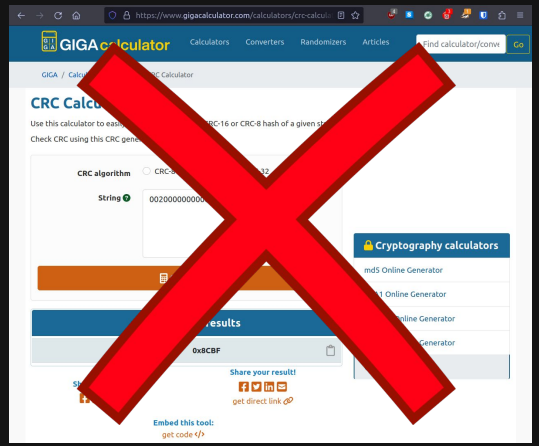


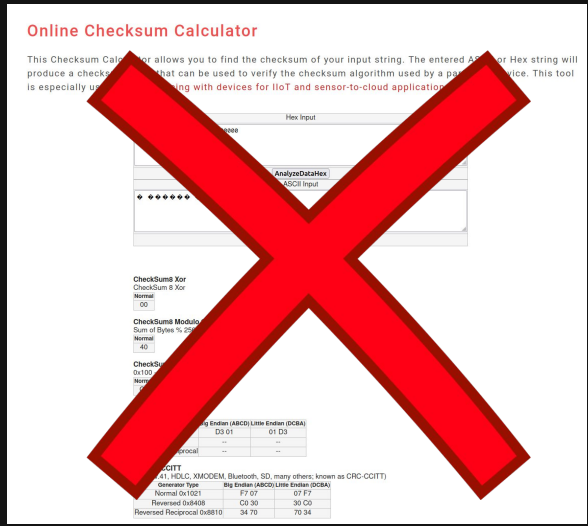


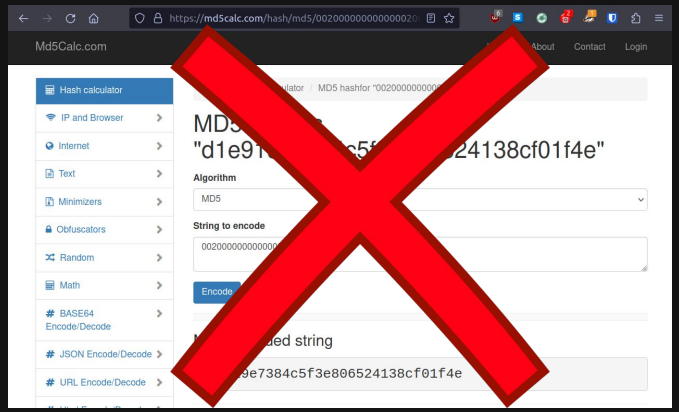


Some common algorithms









What now?

- This checksum is evil
- Maybe cloning cards ain’t so bad
- Maybe I’ll just put the project to rest

November 2021

Then One Day...

February 2022

Let’s figure out where the money is!

Let’s stare at some hex data together!

For hours

```
00000010   4E 0F 04 10  04 10 04 10  04 10 04 10  04 10 04 10  N...............
00000020   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000030   30 60 20 6F  5B 0A 78 77  88 C1 F1 B9  F5 66 9C C8  0` o[.xw.....f..
00000040   04 10 23 45  66 77 00 00  00 00 00 00  00 00 00 00  ..#Efw..........
00000050   00 03 80 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000060   00 02 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000070   5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89  ^.../+xw...b$.~.
00000080   11 89 1C 00  02 90 89 1C  00 65 00 00  00 00 85 9C  .........e......
00000090   5B FB 2C 03  E0 00 00 00  00 20 00 00  00 00 15 2D  [.,...... .....-
000000A0   00 20 00 00  00 00 00 00  00 00 00 00  00 00 28 E3  . ............(.
000000B0   5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89  ^.../+xw...b$.~.
000000C0   11 9A 32 AA  BF F8 9A 32  AA 65 00 00  90 00 1A E3  ..2....2.e......
000000D0   5B FD 4E D9  20 00 32 00  00 20 00 00  00 00 BD 1E  [.N. .2.. ......
000000E0   00 20 00 00  00 00 00 00  00 00 00 00  00 00 28 E3  . ............(.
000000F0   5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89  ^.../+xw...b$.~.
00000100   00 20 00 00  00 00 00 00  20 00 00 00  00 00 88 C3  . ...... .......
```

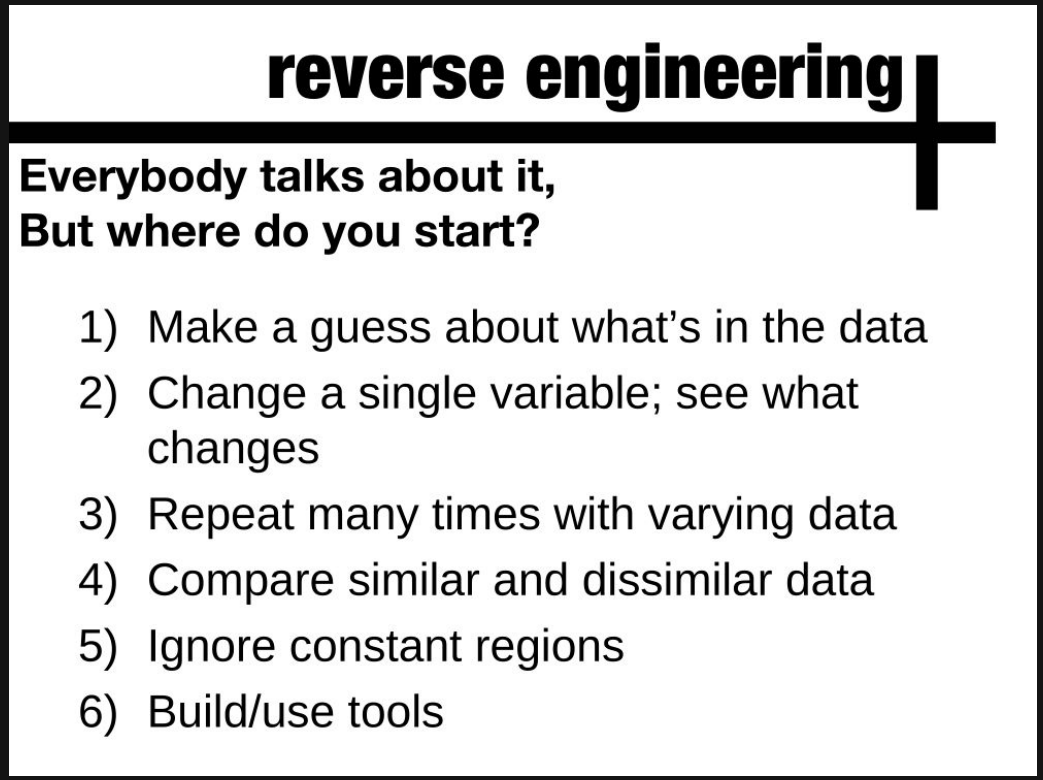
and hours


00000010 4E 0F 04 10 04 10 04 10 04 10 04 10 04 10 04 10 N...............
00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000030 30 60 20 6F 5B 0A 78 77 88 C1 F1 B9 F5 66 9C C8 0` o[.xw.....f..
00000040 04 10 23 45 66 77 00 00 00 00 00 00 00 00 00 00 ..#Efw..........
00000050 00 03 80 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000060 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00000070 5E C3 9B 02 2F 2B 78 77 88 00 F6 62 24 8E 7E 89 ^.../+xw...b$.~.
00000080 11 89 1C 00 02 90 89 1C 00 65 00 00 00 00 85 9C ....e......
00000090 5B FB 2C 03 E0 00 00 00 00 20 00 00 00 00 15 2D [.,...... .....-
000000A0 00 20 00 00 00 00 00 00 00 00 00 00 00 00 28 E3 . ...(.
000000B0 5E C3 9B 02 2F 2B 78 77 88 00 F6 62 24 8E 7E 89 ^.../+xw...b$.~.
000000C0 11 9A 32 AA BF F8 9A 32 AA 65 00 00 90 00 1A E3 ..2....2.e......
000000D0 5B FD 4E D9 20 00 32 00 00 20 00 00 00 00 BD 1E [.N. .2.. ......
000000E0 00 20 00 00 00 00 00 00 00 00 00 00 00 00 28 E3 . ...(.
000000F0 5E C3 9B 02 2F 2B 78 77 88 00 F6 62 24 8E 7E 89 ^.../+xw...b$.~.
00000100 00 20 00 00 00 00 00 00 20 00 00 00 00 00 88 C3 . ...

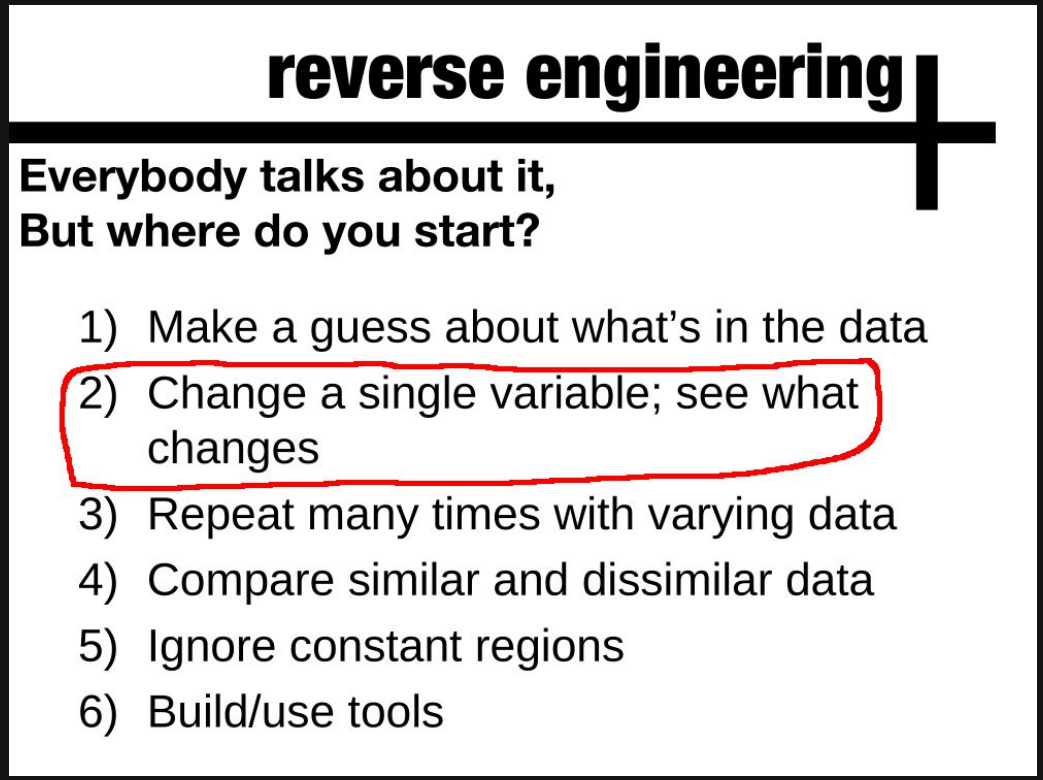

2000 YEARS
LATER

We Found Nothing

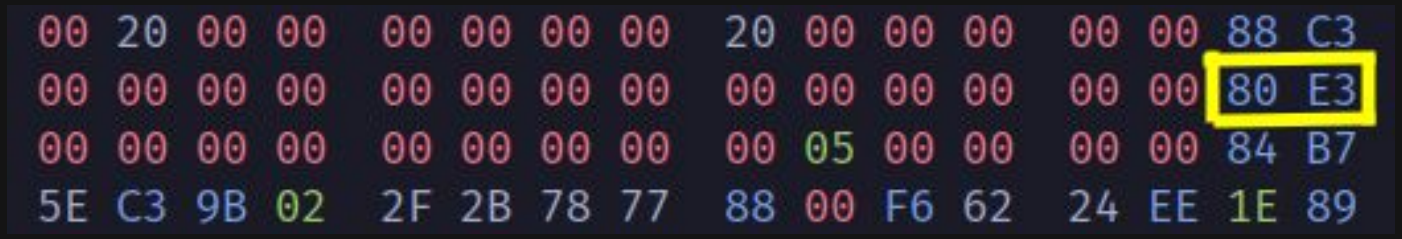
How do you reverse engineer things again?

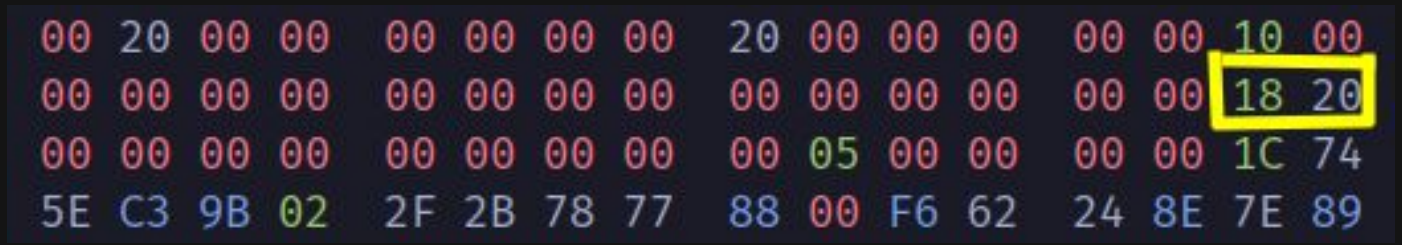


How do you reverse engineer things again?

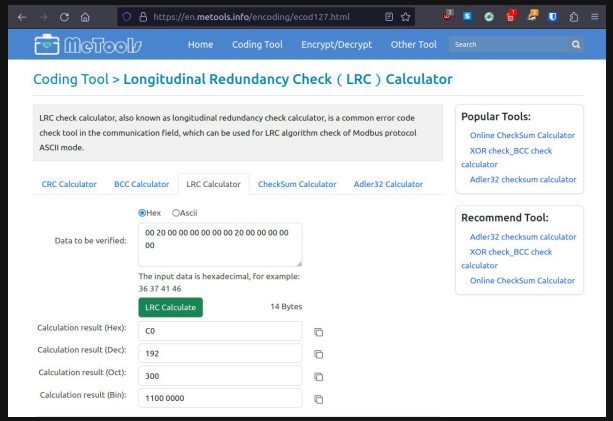


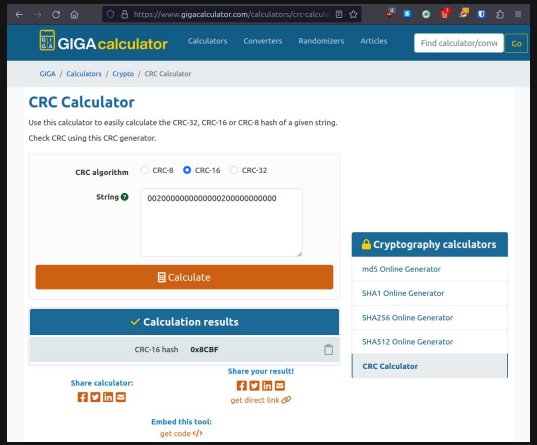
Maybe Let’s Try Another Crack at that Checksum

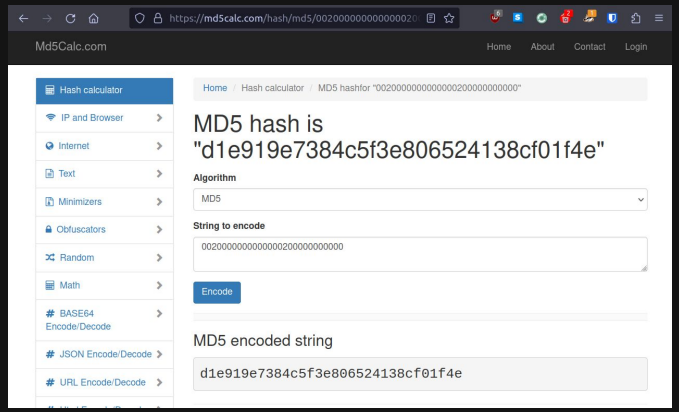

00 20 00 00 00 00 00 00 20 00 00 00 00 00 88 C3
00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 E3
00 00 00 00 00 00 00 00 00 05 00 00 00 00 84 B7
5E C3 9B 02 2F 2B 78 77 88 00 F6 62 24 EE 1E 89

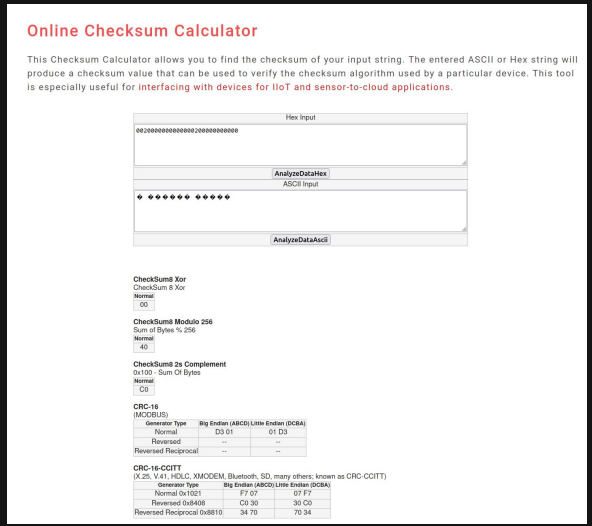

00 20 00 00 00 00 00 00 20 00 00 00 00 00 10 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 18 20
00 00 00 00 00 00 00 00 00 05 00 00 00 00 1C 74
5E C3 9B 02 2F 2B 78 77 88 00 F6 62 24 8E 7E 89

Maybe It's an Off The Shelf Checksum… With Salt

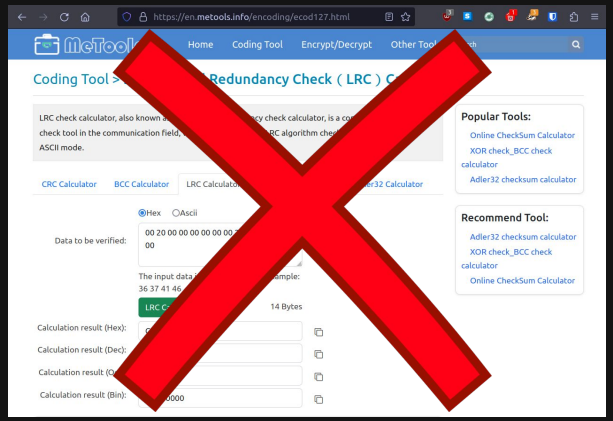


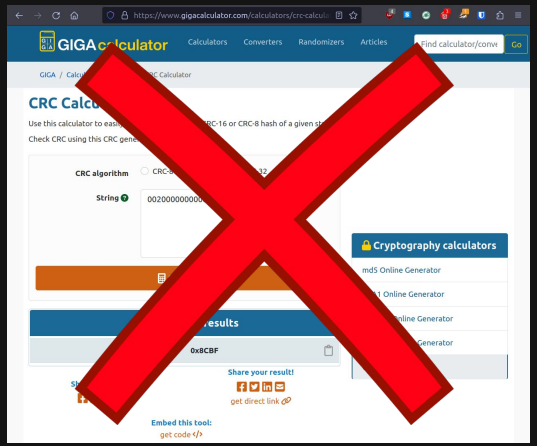


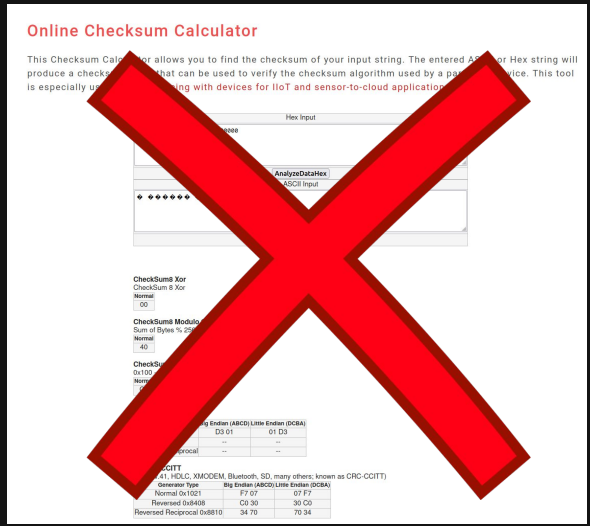


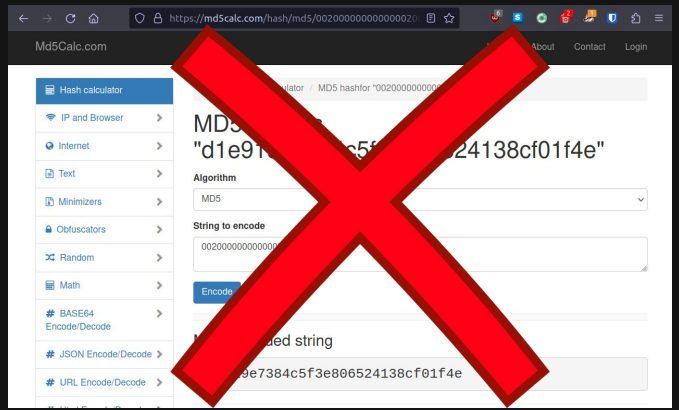


Or not


https://en.metools.info/encoding/ecod127.html
Home
Coding Tool
Encrypt/Decrypt
Other Tool
Coding Tool >
Redundancy Check (LRC) C
Popular Tools:
Online CheckSum Calculator
XOR check_BCC check calculator
Adler32 checksum calculator
Recommend Tool:
Adler32 checksum calculator
XOR check_BCC check calculator
Online CheckSum Calculator
CRC Calculator
BCC Calculator
Hex
Ascii
Data to be verified:
14 Bytes
Calculation result (Hex):
Calculation result (Dec):
Calculation result (Bin):


GIGA calculator
Calculators
Converters
Randomizers
Articles
CRC Calc
CRC algorithm
String
Cryptography calculators
md5 Online Generator
0x8CBF
Share your result!
get direct link
Embed this tool:
get code


Online Checksum Calculator
Hex Input
AnalyzeDataHex
ASCII Input
CheckSum8 Xor
CheckSum 8 Xor
Normal
00
CheckSum8 Modulo
Sum of Bytes % 256
Normal
40
Big Endian (ABCD)
Little Endian (DCBA)
D3 01
01 D3
CCITT
Generator Type
Normal 0x1021
Reversed 0x8408
Reversed Reciprocal 0x8810
F7 CF
C0 30
34 70
CF F7
30 C0
70 34

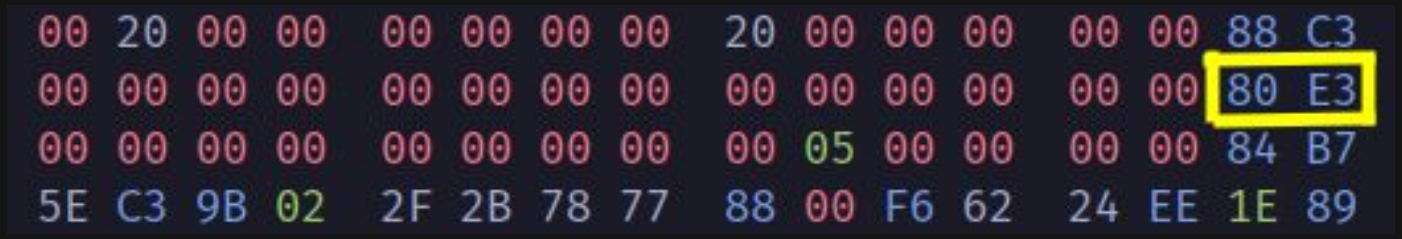

https://md5calc.com/hash/md5/
Md5Calc.com
About
Contact
Login
Hash calculator
IP and Browser
Internet
Text
Minimizers
Obfuscators
Random
Math
BASE64 Encode/Decode
JSON Encode/Decode
URL Encode/Decode
Algorithm
MD5
String to encode
Encode
"d1e91
24138cf01f4e"
ed string
9e7384c5f3e806524138cf01f4e

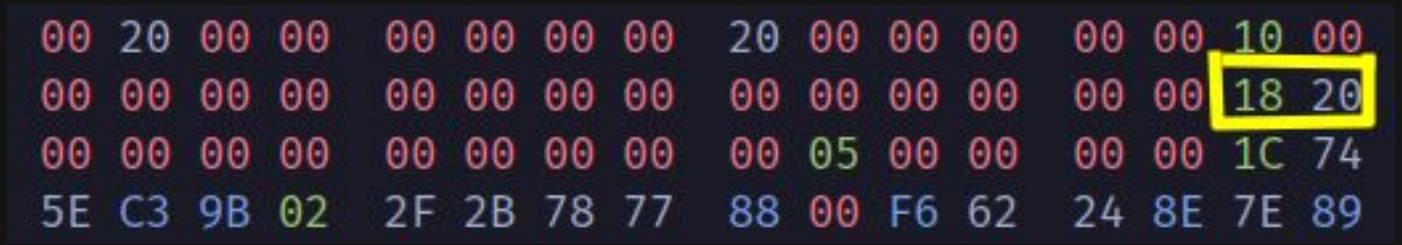
~~English Class~~ CharlieCard Hacking Class

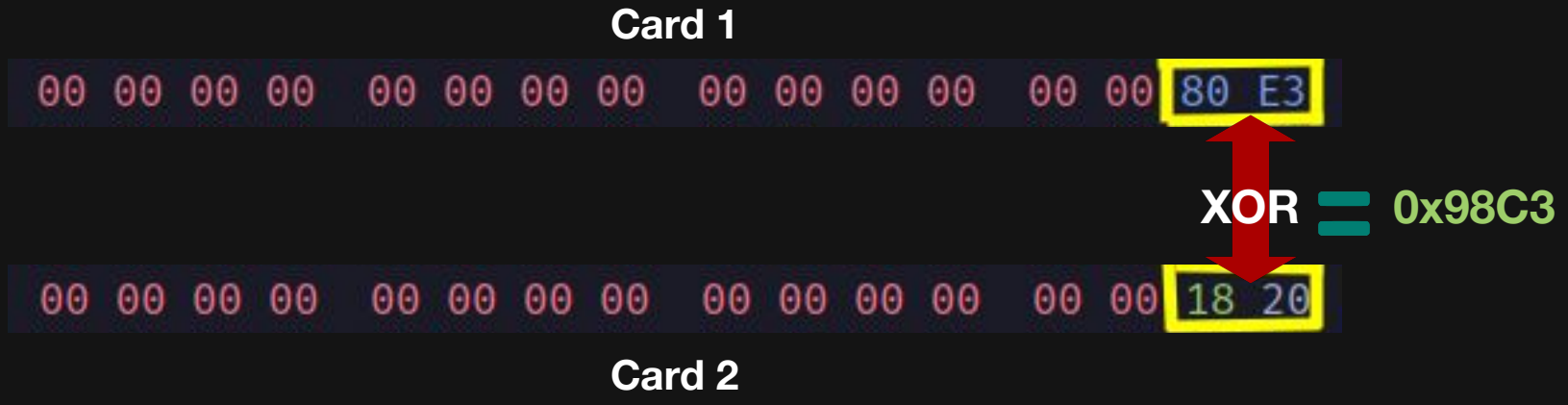
- What to do during English class
- Learn… No
- Work… No
- Stare at hex data and try random checksum algorithms… Yes!

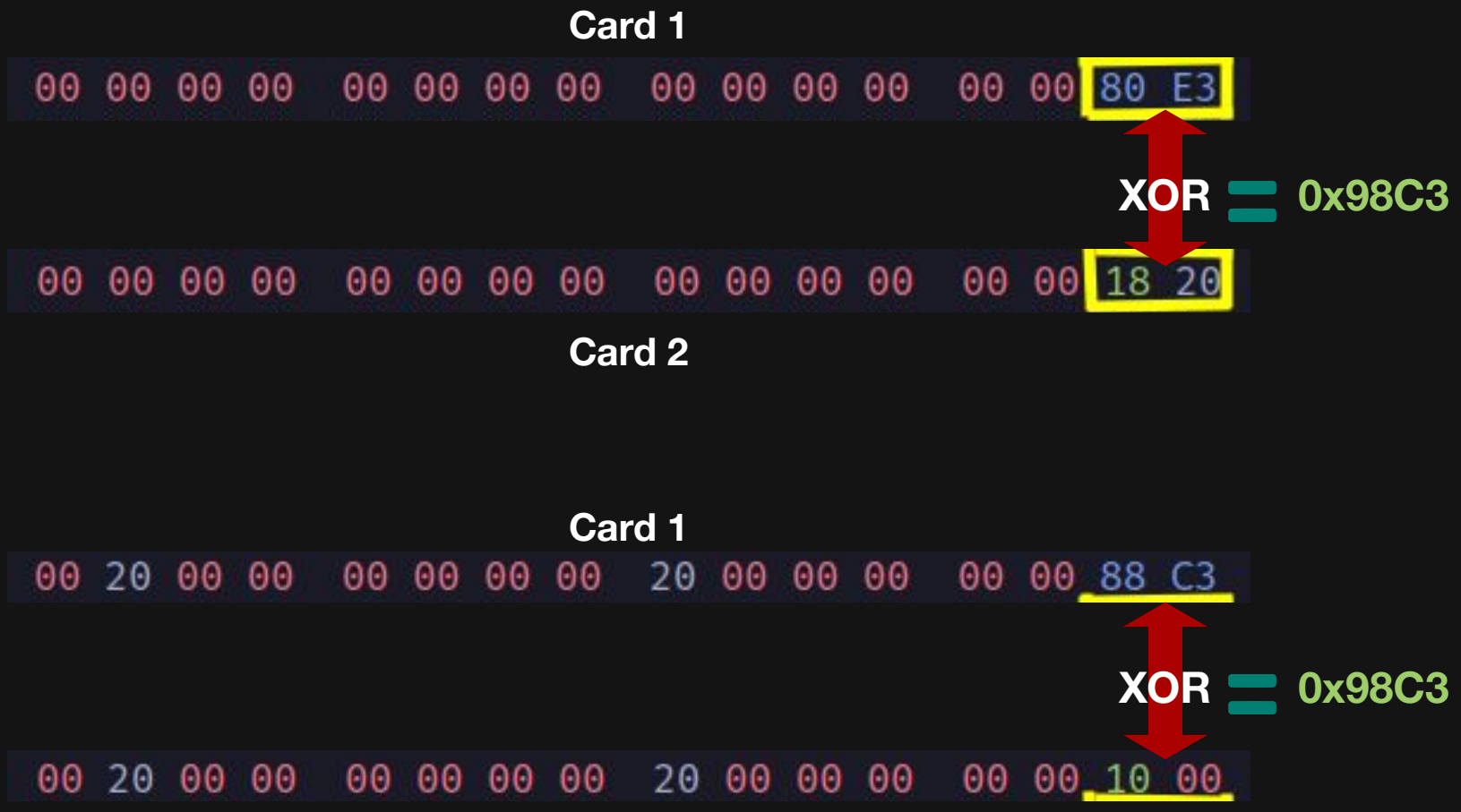
Our First Breakthrough

March 2022


00 20 00 00 00 00 00 00 20 00 00 00 00 00 88 C3
00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 E3
00 00 00 00 00 00 00 00 00 05 00 00 00 00 84 B7
5E C3 9B 02 2F 2B 78 77 88 00 F6 62 24 EE 1E 89


00 20 00 00 00 00 00 00 20 00 00 00 00 00 10 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 18 20
00 00 00 00 00 00 00 00 00 05 00 00 00 00 1C 74
5E C3 9B 02 2F 2B 78 77 88 00 F6 62 24 8E 7E 89


Card 1
00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 E3
XOR
0x98C3
00 00 00 00 00 00 00 00 00 00 00 00 00 00 18 20
Card 2


Card 1
00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 E3
XOR = 0x98C3
00 00 00 00 00 00 00 00 00 00 00 00 00 00 18 20
Card 2
Card 1
00 20 00 00 00 00 00 00 20 00 00 00 00 00 88 C3
XOR = 0x98C3
00 20 00 00 00 00 00 00 20 00 00 00 00 00 10 00
Card 2

So How Do We Use This?

Find Two Identical Lines

```
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 88 C3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 05 00 00  00 00 84 B7
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  05 00 00 9A  32 AA BF FD  00 32 9A 53
9A 32 AA BF  FD 00 32 00  00 00 00 00  00 00 FF 11
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF
FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF
```

```
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 10 00
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
00 00 00 00  00 00 00 00  00 05 00 00  00 00 1C 74
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  05 00 00 9E  37 AF C0 8D  00 32 92 23
9E 37 AF C0  8D 00 32 00  00 00 00 00  00 00 7E 9A
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF
FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF
```

- Find two lines with the same data on two different cards
- Note that identical lines on different cards have a different checksum

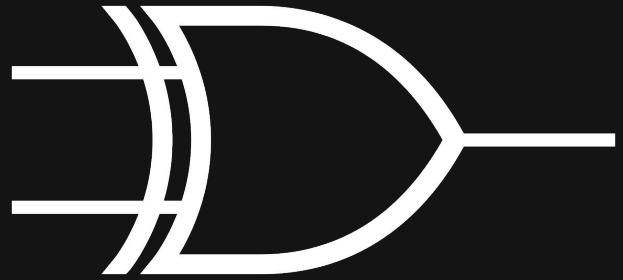
Find Checksum Modifier

```
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 88 C3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 05 00 00  00 00 84 B7
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  05 00 00 9A  32 AA BF FD  00 32 9A 53
9A 32 AA BF  FD 00 32 00  00 00 00 00  00 00 FF 11
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF
FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF
```

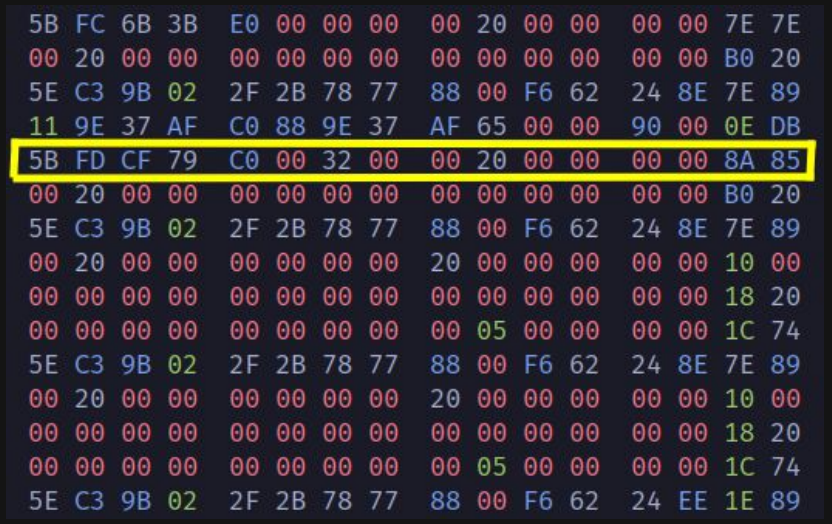
```
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 10 00
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
00 00 00 00  00 00 00 00  00 05 00 00  00 00 1C 74
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 10 00
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
00 00 00 00  00 00 00 00  00 05 00 00  00 00 1C 74
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  05 00 00 9E  37 AF C0 8D  00 32 92 23
9E 37 AF C0  8D 00 32 00  00 00 00 00  00 00 7E 9A
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
00 00 00 00  00 00 00 00  00 00 00 00  00 00 18 20
```

- XOR the checksums
- The value you get is the “checksum modifier” between the two cards

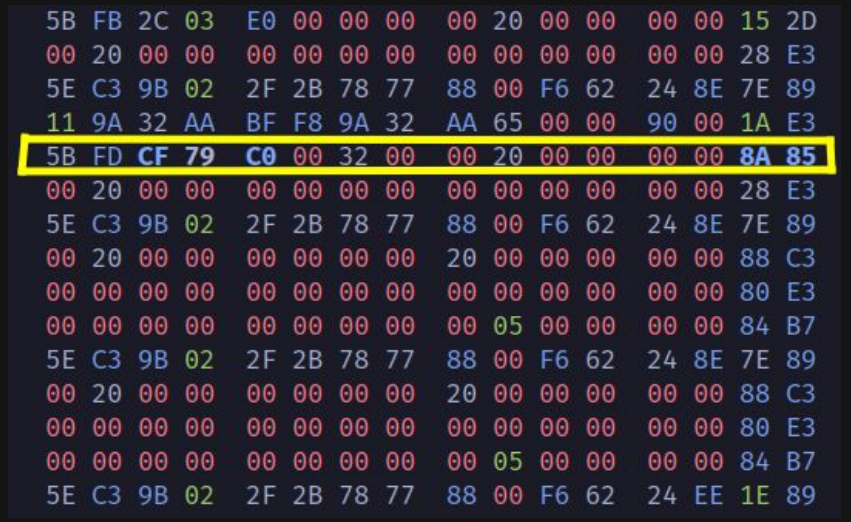
```
  80E3
XOR 1820
  98C3
```



Copy Any Line From Old Card to New Card



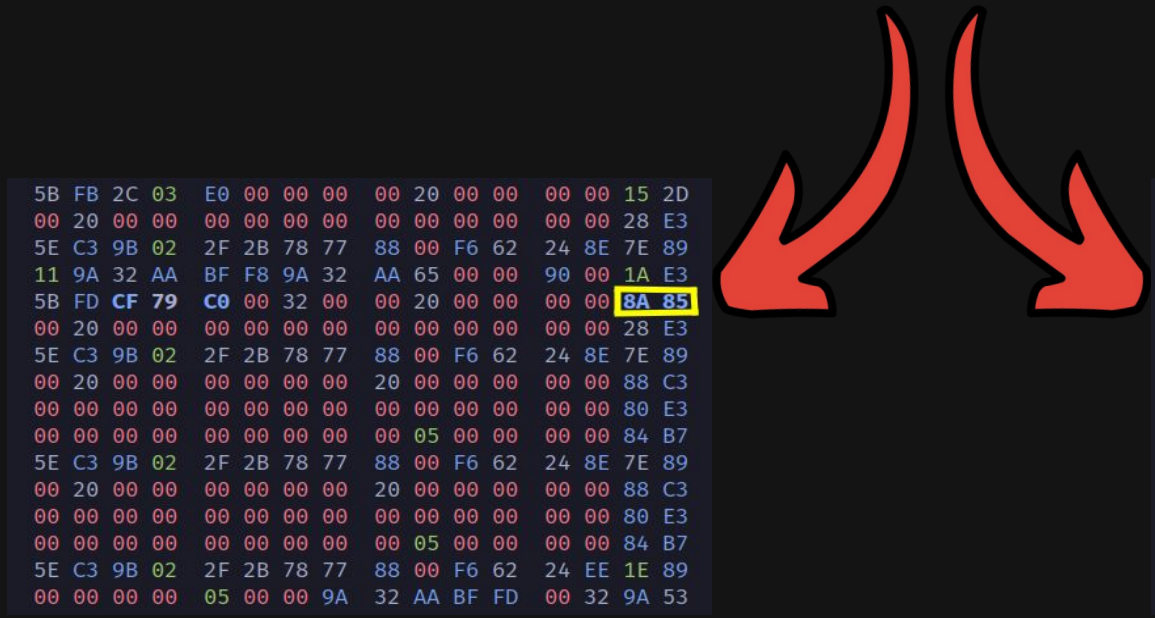
Card 1



Card 2

XOR Checksum With Checksum Modifier To Get New Checksum

0x8A85 XOR 0x98C3 = 0x1246



```
5B FB 2C 03  E0 00 00 00  00 20 00 00  00 00 15 2D
00 20 00 00  00 00 00 00  00 00 00 00  00 00 28 E3
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
11 9A 32 AA  BF F8 9A 32  AA 65 00 00  90 00 1A E3
5B FD CF 79  C0 00 32 00  00 20 00 00  00 00 8A 85
00 20 00 00  00 00 00 00  00 00 00 00  00 00 28 E3
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 88 C3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 05 00 00  00 00 84 B7
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 88 C3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 05 00 00  00 00 84 B7
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
00 00 00 00  05 00 00 9A  32 AA BF FD  00 32 9A 53
```

```
5B FB 2C 03  E0 00 00 00  00 20 00 00  00 00 15 2D
00 20 00 00  00 00 00 00  00 00 00 00  00 00 28 E3
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
11 9A 32 AA  BF F8 9A 32  AA 65 00 00  90 00 1A E3
5B FD CF 79  C0 00 32 00  00 20 00 00  00 00 12 46
00 20 00 00  00 00 00 00  00 00 00 00  00 00 28 E3
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 88 C3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 05 00 00  00 00 84 B7
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 8E 7E 89
00 20 00 00  00 00 00 00  20 00 00 00  00 00 88 C3
00 00 00 00  00 00 00 00  00 00 00 00  00 00 80 E3
00 00 00 00  00 00 00 00  00 05 00 00  00 00 84 B7
5E C3 9B 02  2F 2B 78 77  88 00 F6 62  24 EE 1E 89
```

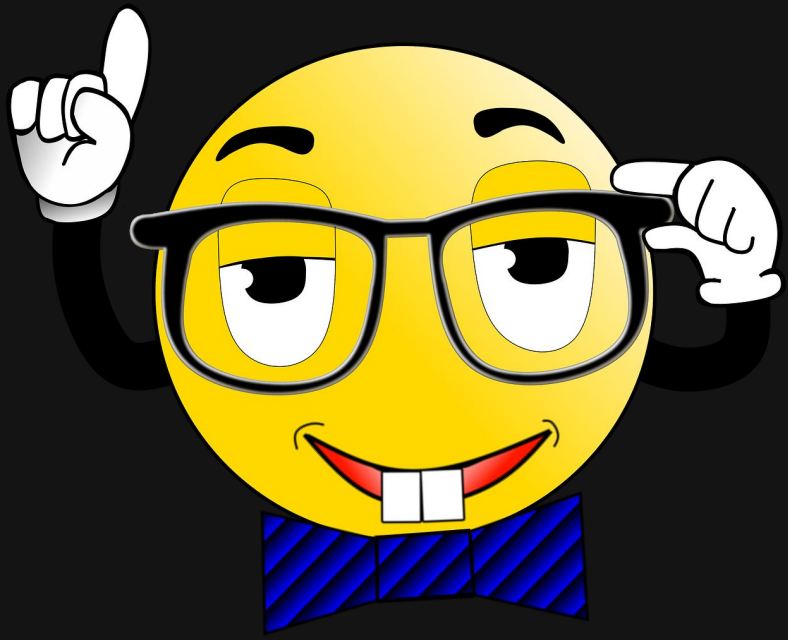
BOOM! LINE = COPIED

Instructions

- Step 1: Start with two cards
- Step 2: Find two identical lines
- Step 3: XOR the two different checksums to get checksum modifier
- Step 4: Find a line you want to move
- Step 5: Copy that line onto the other card
- Step 6: XOR the old checksum by checksum modifier
- Step 7: Profit

What can we do now?

- Copy money values from card to card
- Copy anything from card to card
- Better than cloning, can’t get disabled



Next step: Trial and error

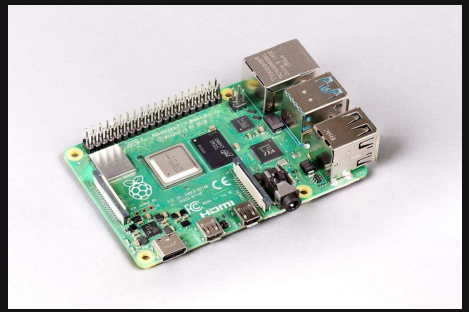
Subway Station Adventure Time!

March 2022

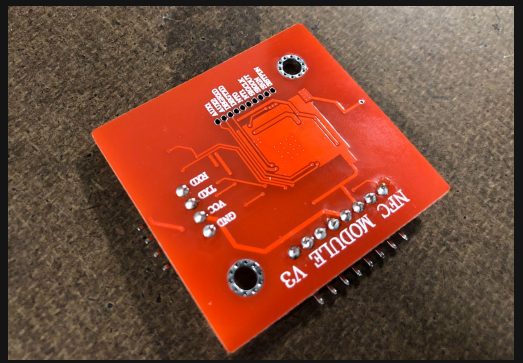
Step 1: Make an NFC Reader Contraption

- Ingredients: PN532, Raspberry Pi, Battery Bank
- Step 1: Hook up PN532 to Raspberry Pi
- Step 2: Plug battery bank into Raspberry Pi
- Step 3: Make your own WiFi hotspot and SSH into the Raspberry Pi
- Step 4: Toss contraption in backpack

+



+




STUDENT
CharlieCard

Step 2: Spend ~~hours~~ days at a train station

Both Stations Smell Funny

Step 3: Mess around

- Make ~~random~~ educated guess-based changes
- Flash changes onto cards
- Tap cards
- Enjoy error noise
- Try to not make it obvious you’re up to no good

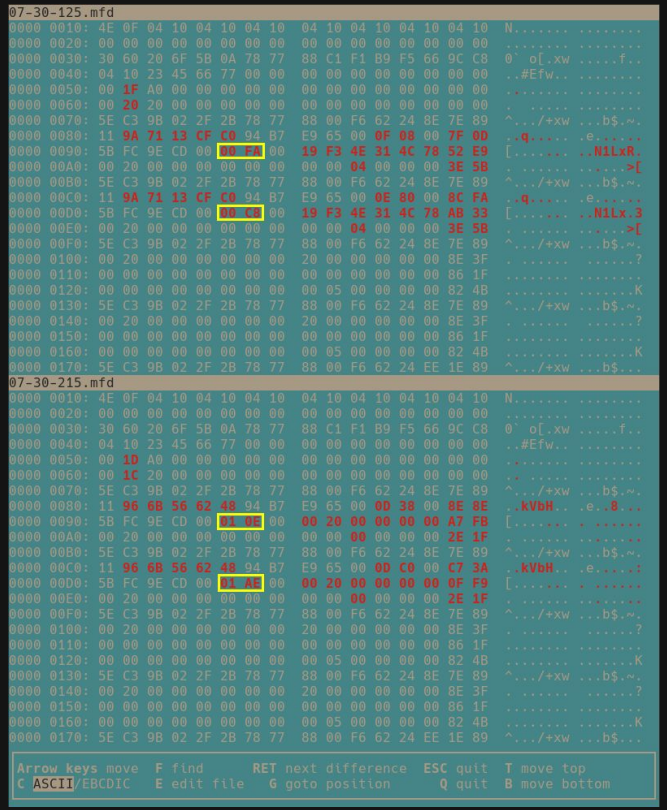
Another Breakthrough

March 2022

Take A Look At These Bytes

Current Value:
$1.25
Last Value:
$1.00

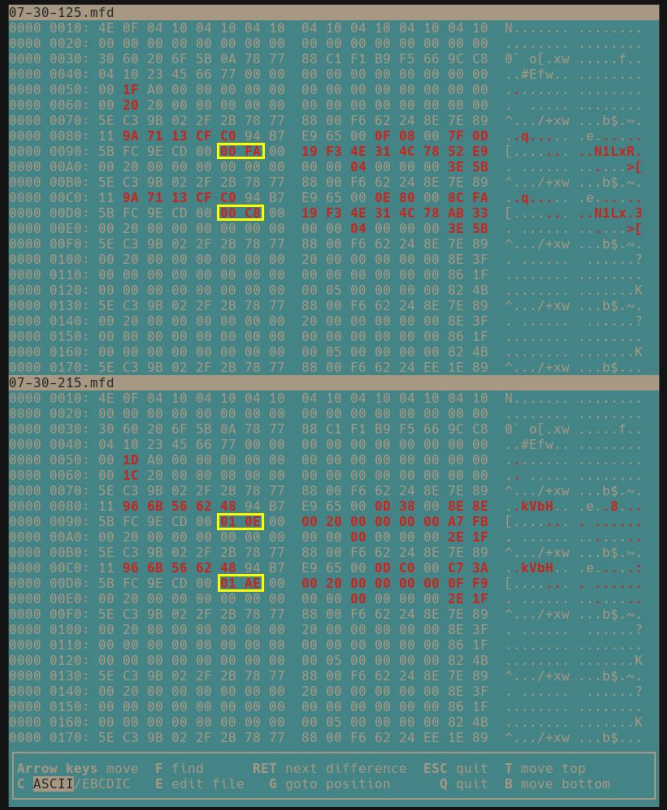
Current Value:
$2.15
Last Value:
$1.35



Convert From Hex to Decimal

Current Value:
$1.25
Last Value:
$1.00

Current Value:
$2.15
Last Value:
$1.35



0xFA -> 250
0xC8 -> 200

0x010E -> 270
0x01AE -> 430

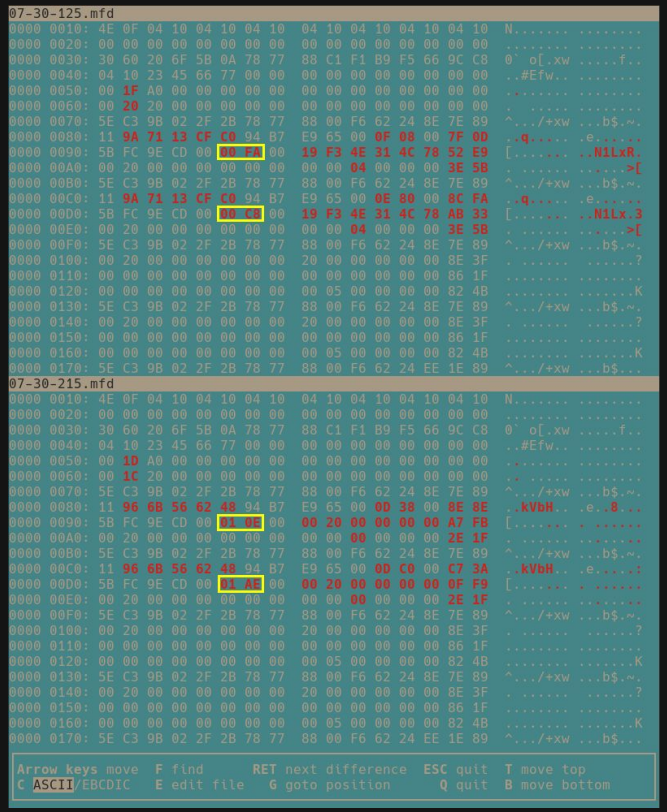
Interesting…

Current Value:
$1.25
Last Value:
$1.00

0xFA -> 250 / 2 -> 125
0xC8 -> 200 / 2 -> 100

Current Value:
$2.15
Last Value:
$1.35

0x010E -> 270 / 2 -> 135
0x01AE -> 430 / 2 -> 215



Our Findings

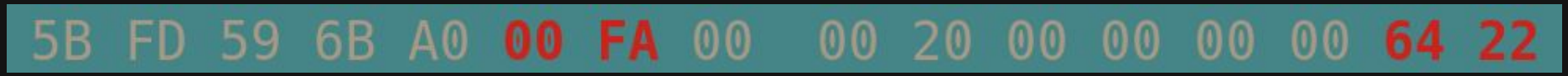
- Where the money's stored!!!
- Money stored in half-pennies
- Two transaction registers (current and last money value)
- CharlieCards are weird

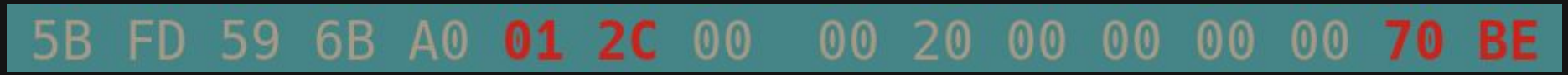
Let’s Try to Change It

April 2022

Step 1: Isolate variables

0xFA = 250 half pennies = $1.25

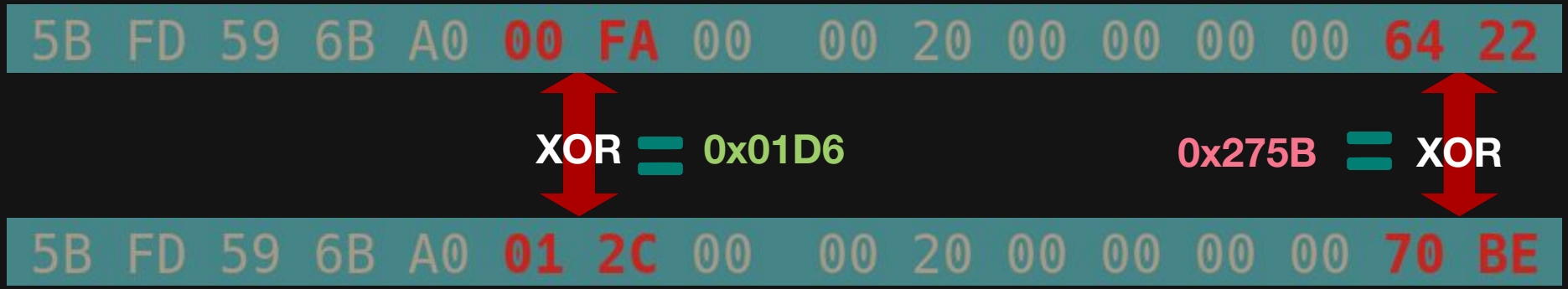




0x012C = 300 half pennies = $1.50

Step 2: XOR everything

0xFA = 250 half pennies = $1.25



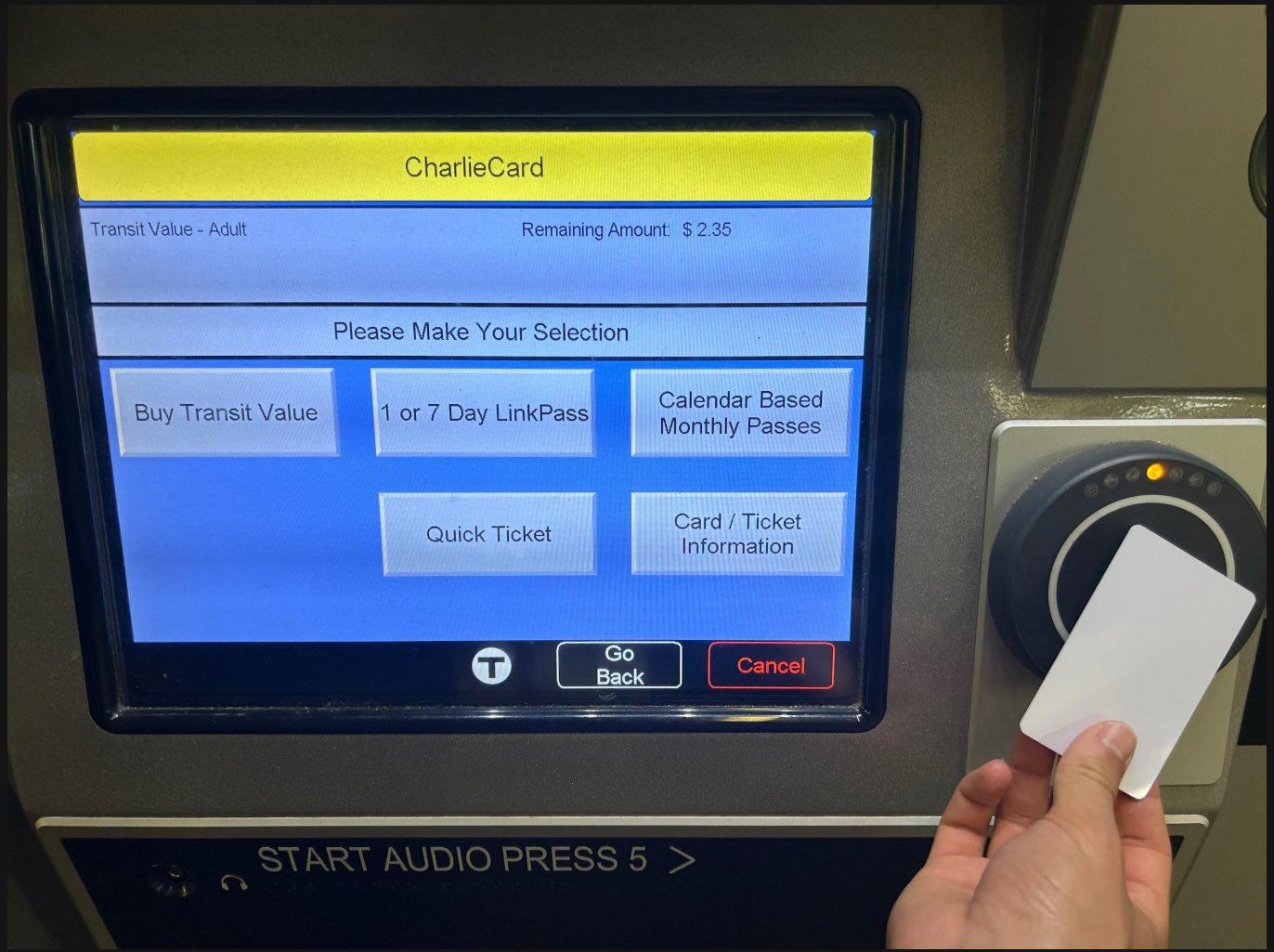
0x012C = 300 half pennies = $1.50

Data Modifier
Checksum Modifier

Step 3: Try this strategy with different money values

0x0 = 0 half pennies = $0.00

0x01D6 = 470 half pennies = $2.35


CharlieCard
Transit Value - Adult
Remaining Amount: $ 2.35
Please Make Your Selection
Buy Transit Value
1 or 7 Day LinkPass
Calendar Based Monthly Passes
Quick Ticket
Card / Ticket Information
Go Back
Cancel
START AUDIO PRESS 5 >

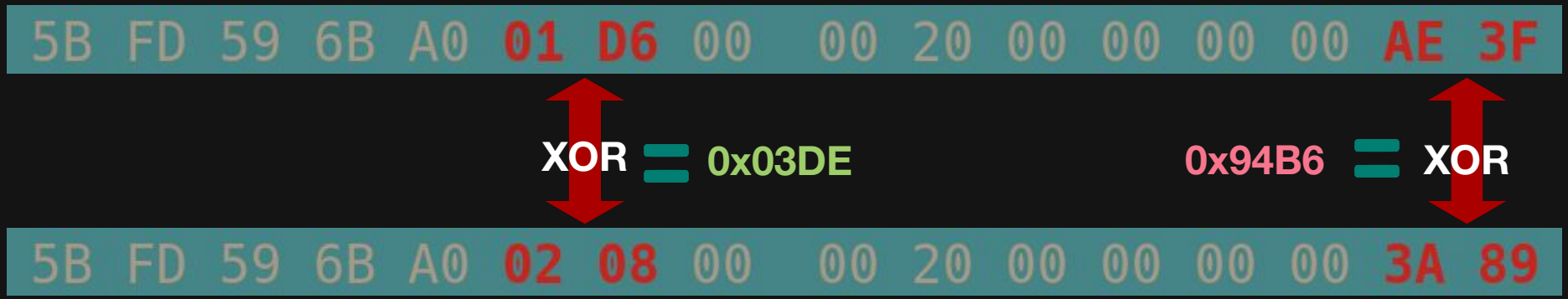
Step 4: Add a quarter

Why add a quarter?

```
0b0000_1000 XOR 0b0000_0111 = 0b0000_1111
   (8)             (7)          (15)

0b0000_1111  +       1      = 0b0001_0000
   (15)                         (16)

0b0001_0000 XOR 0b0000_1111 = 0001_1111
   (16)             (15)         (31)
```

Step 5: Inch your way up

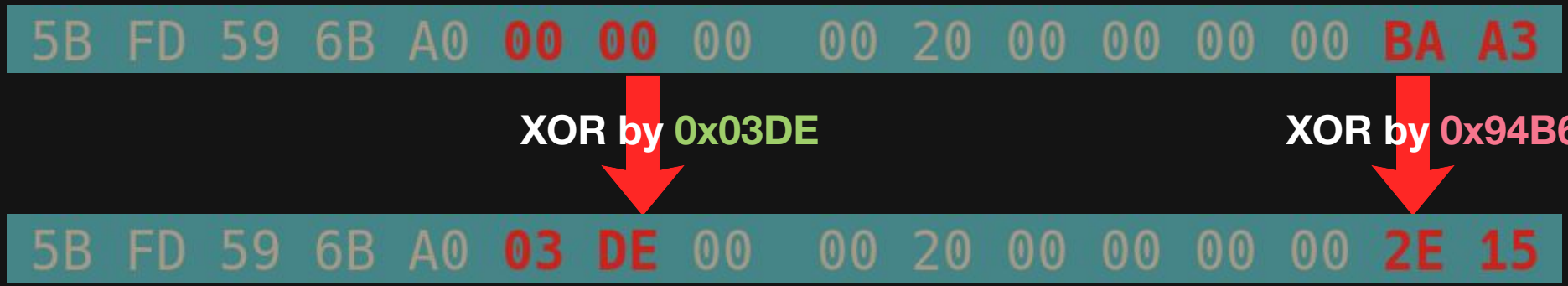
0x01D6 = 470 half pennies = $2.35



0x0208 = 520 half pennies = $2.60

Step 5: Inch your way up

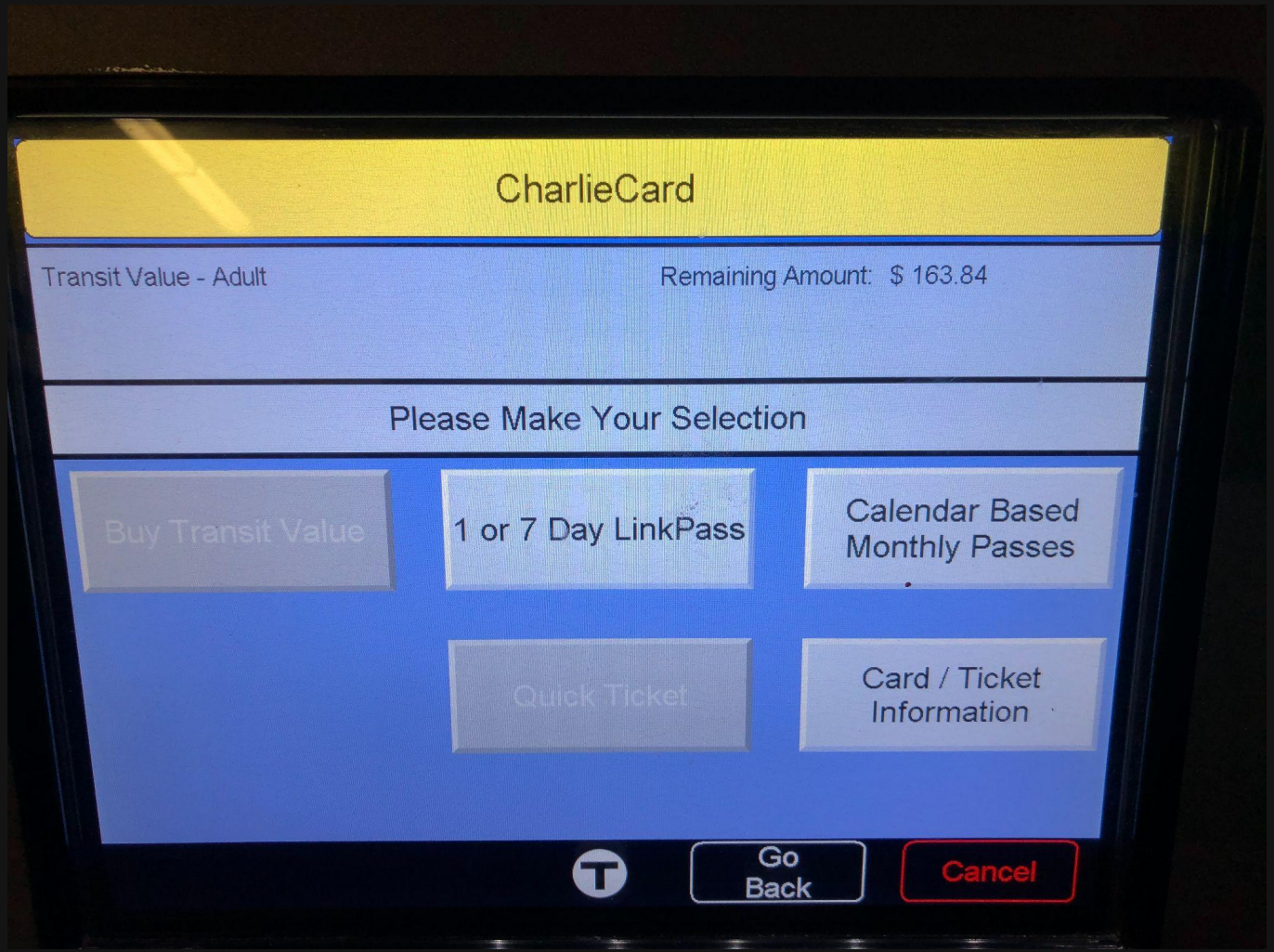
0x00 = 0 half pennies = $0.00

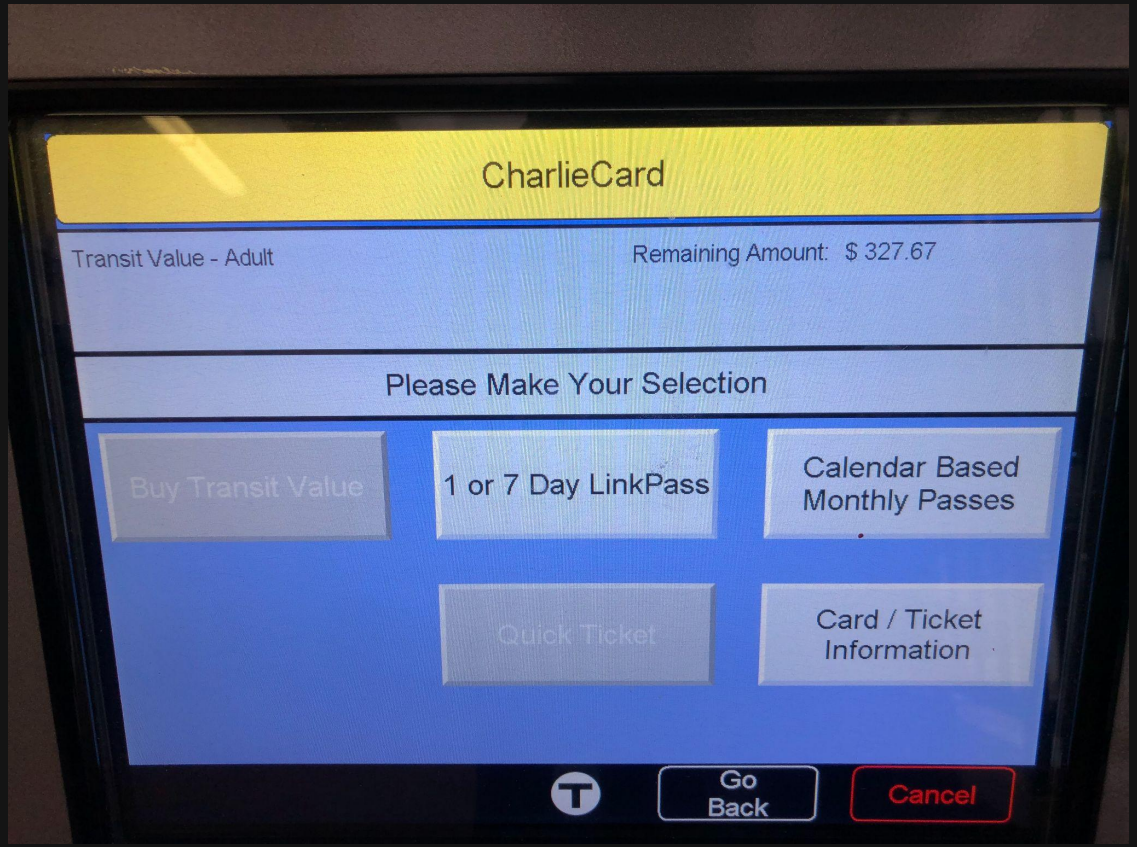


0x03DE = 990 half pennies = $4.95


CharlieCard
Transit Value - Adult
Remaining Amount: $ 4.95
Please Make Your Selection
Buy Transit Value
1 or 7 Day LinkPass
Calendar Based Monthly Passes
Quick Ticket
Card / Ticket Information
Go Back
Cancel
START AUDIO PRESS 5
TICKETS / CHANGE

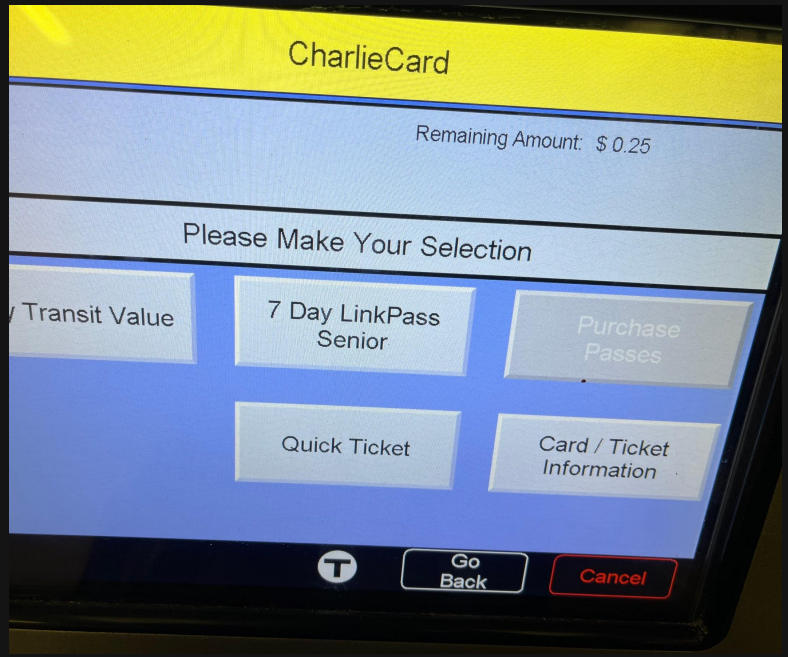
Rinse and Repeat


CharlieCard
Transit Value - Adult
Remaining Amount: $ 163.84
Please Make Your Selection
Buy Transit Value
1 or 7 Day LinkPass
Calendar Based Monthly Passes
Quick Ticket
Card / Ticket Information
Go Back
Cancel


CharlieCard
Transit Value - Adult
Remaining Amount: $ 327.67
Please Make Your Selection
Buy Transit Value
1 or 7 Day LinkPass
Calendar Based Monthly Passes
Quick Ticket
Card / Ticket Information
Go Back
Cancel

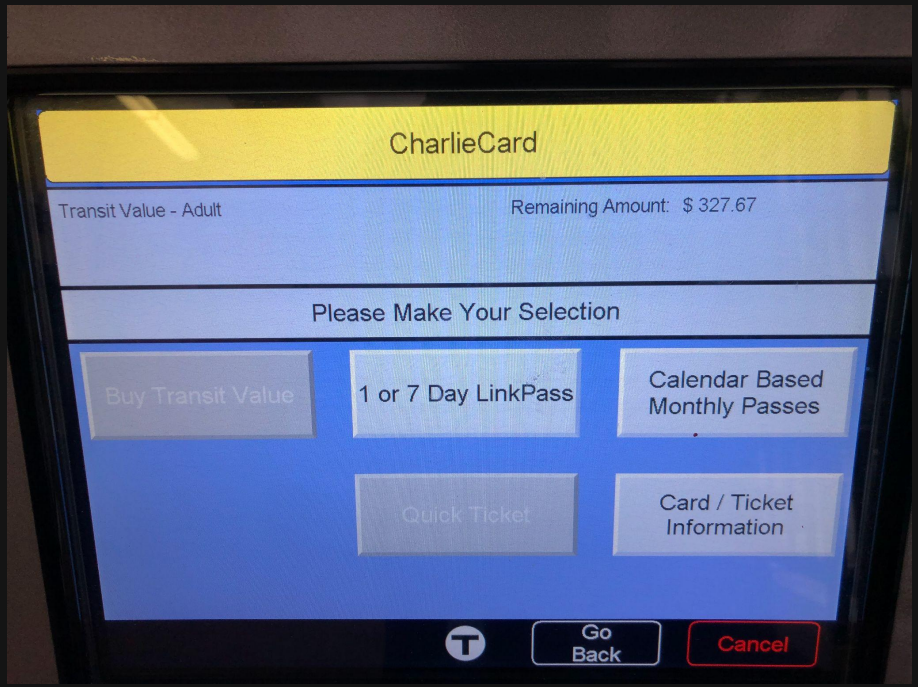
Free rides for life: A free step process

- **Step 1: Buy a card for $0.25**



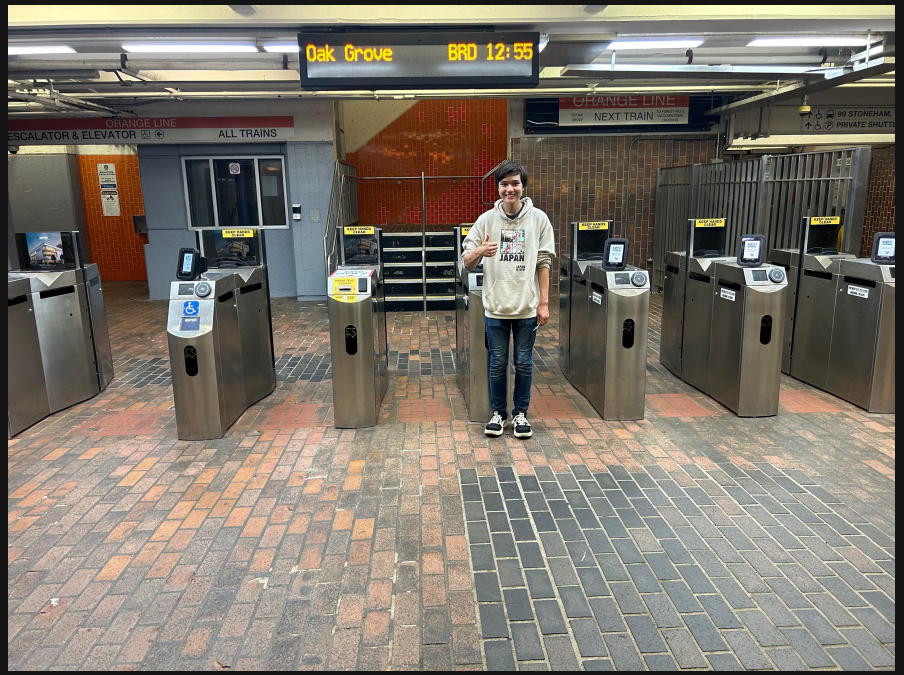
Free rides for life: A free step process

- Step 1: Buy a card for $0.25
- Step 2: Set the value to $327.67



Free rides for life: A free step process

- Step 1: Buy a card for $0.25
- Step 2: Set the value to $327.67
- Step 3: Profit!

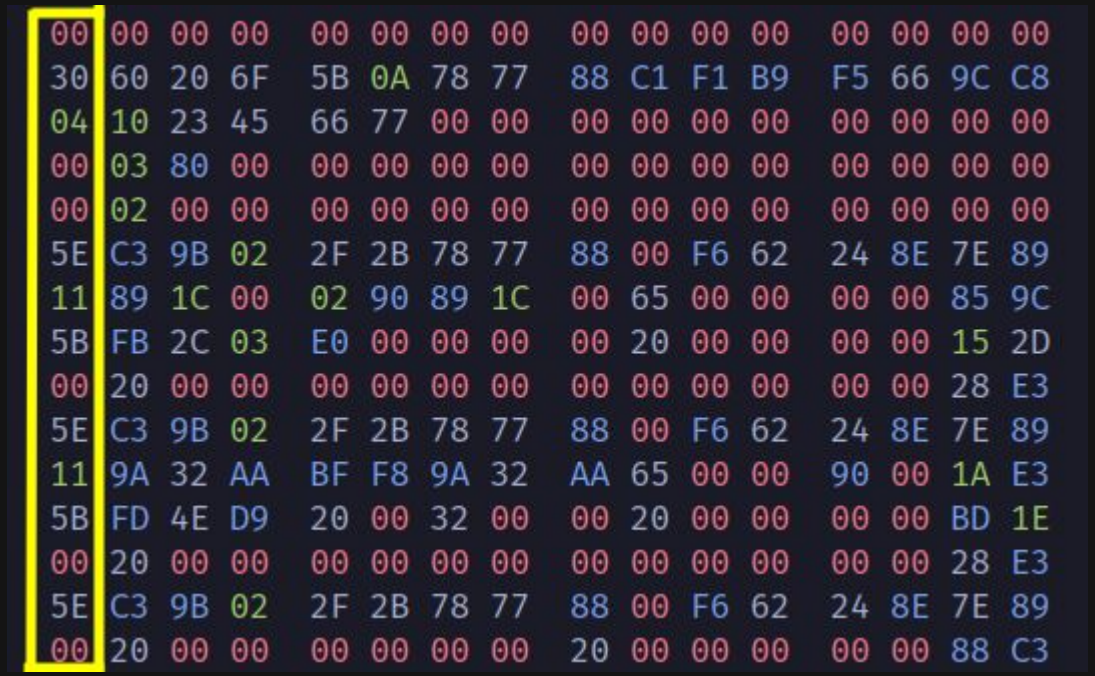


How Does This Work?

Vocab

1. Existing Data -> Pre-existing data that you want to edit
2. Target Data -> Value you want the data to be
3. Data Modifier -> Value you need to XOR data by to get Target Data Value
4. Existing Checksum -> You know what this is
5. Target Checksum -> Value you need to make the Target Data valid
6. Checksum Modifier -> Value you need to XOR the checksum with to make it valid
7. Look-Up Table -> A super-secret black box that takes in a Data Modifier and a column and returns a Checksum Modifier

What's a column?



Existing Data XOR Target Data = Data Modifier

Existing Checksum XOR Checksum Modifier = Target Checksum

Example With Steps

1. Choose What to Change: We Want to Change 0x32 ($0.25) to 0xC8 ($1) in Column 7
2. Find Data Modifier: XOR 0x32 with 0xC8 getting 0xFA
3. Find Checksum Modifier: Put 0xFA and Column 7 into the Lookup Table
4. XOR Checksum Modifier (0xDE81) with the Existing Checksum to get the Target Checksum

A few more station adventures later…

Anatomy of a CharlieCard

UID

Number of Uses

Money

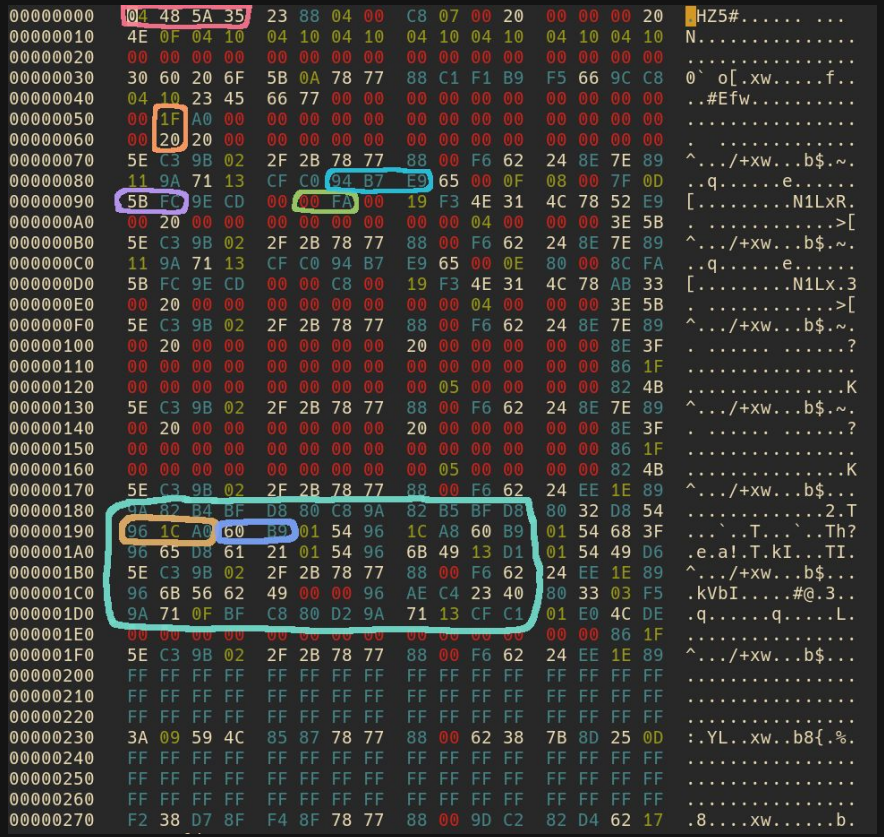
Expiration Date

Card Type

Transaction History

Timestamp

Location



Part 4: Fun Gadgets

Then One Day

April 2022

Vending Machine v1

- Based off the readers on busses
- Fully modeled in CAD
- An abomination
- Impossible for us to manufacture

April 2022

Vending Machine v2

- Still based off of MBTA bus readers
- Fully modeled in CAD *and cardboard*
- Made with manufacturing in mind
- Hard to manufacture
- Couldn't figure out the *slant*

July 2022

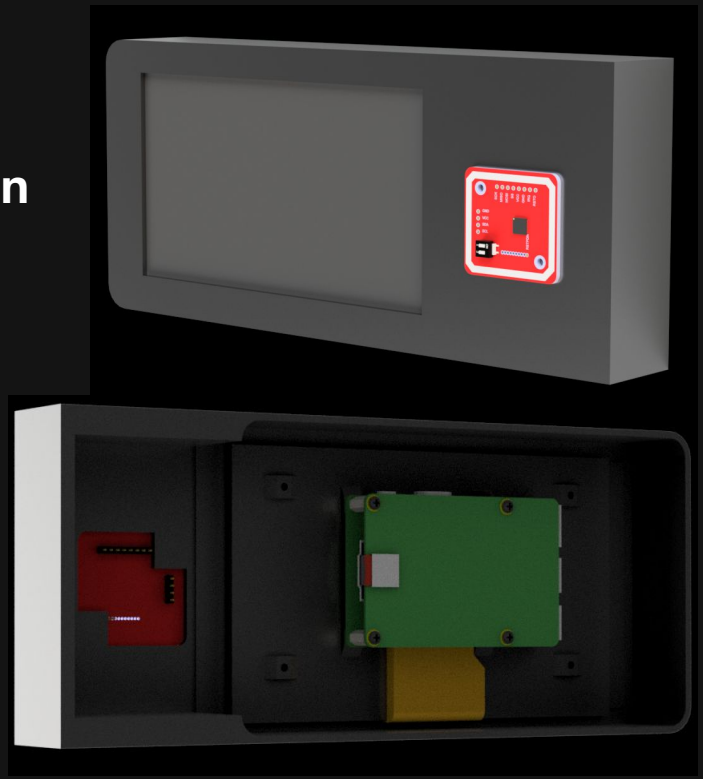
Vending Machine v3

- Still trying to look like bus readers
- Used 3D printed connectors
- Beats the *slant*
- ugly.

September 2022

Vending Machine v4

- The bus readers are ugly anyway
- Slim
- Larger and more responsive touchscreen
- Giant waste of filament
- Impossible to actually mount anything

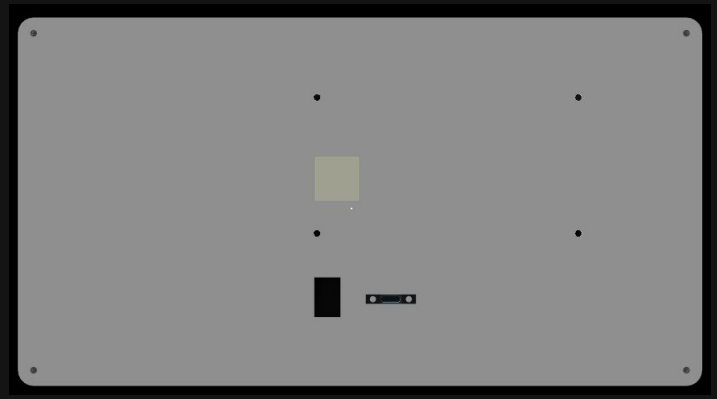


January 2023

Vending Machine v5 (CharlieKiosk)

- Screen mounted upside down :(
- Actually possible to mount parts???
- Sexy
- Baller speaker system
- Couldn't mount usb-c :(

I broke it



February 2023

Vending Machine v6 (CharlieKiosk v2)

- We did it!!!!!
- Screen isn’t upside down
- Mounting ports on the side
- Sleek all black design
- Custom internal mounting :)

March 2023

Vending Machine v6 (CharlieKiosk v2)

- We did it!!!!!
- Screen isn’t upside down
- Mounting ports on the side
- Sleek all black design
- Custom internal mounting :)

I broke it (again)

March 2023

Ingredients

- Raspberry Pi 4
- PN532 NFC reader
- Raspberry Pi Touchscreen
- Adafruit Enclosed Speaker Set
- PLA chassis + bracket
- Acrylic front + back panels
- Usb-c port
- On/off switch
- Blood
- Sweat
- ***Tears***

Then One Day

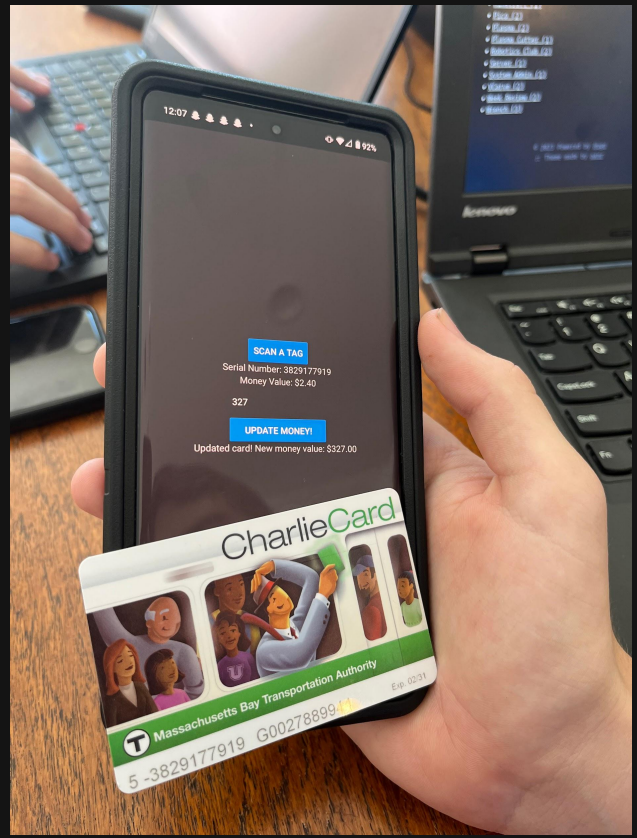
Software

- Vending machine UI made using Tauri
- Charlie Forge library on the backend
- NFC connection on a Rust backend
- Solid JS frontend
- Users can update date, money, expiration date, and card type



Android App

- Slapped together in 2 nights
- React Native
- Ugly



February 2023

Part 5: Not Ending Up In Handcuffs


Chance
GO DIRECTLY
TO JAIL
DO NOT PASS GO, DO NOT COLLECT $200
Jail
Not Jail
Chance
THIS CARD MAY BE KEPT
UNTIL NEEDED OR SOLD
GET OUT OF JAIL
FREE

Your CharlieCard can be hacked by an Android phone, MBTA admits

By Hiawatha Bray Globe Staff, Updated December 8, 2022, 1:28 p.m.

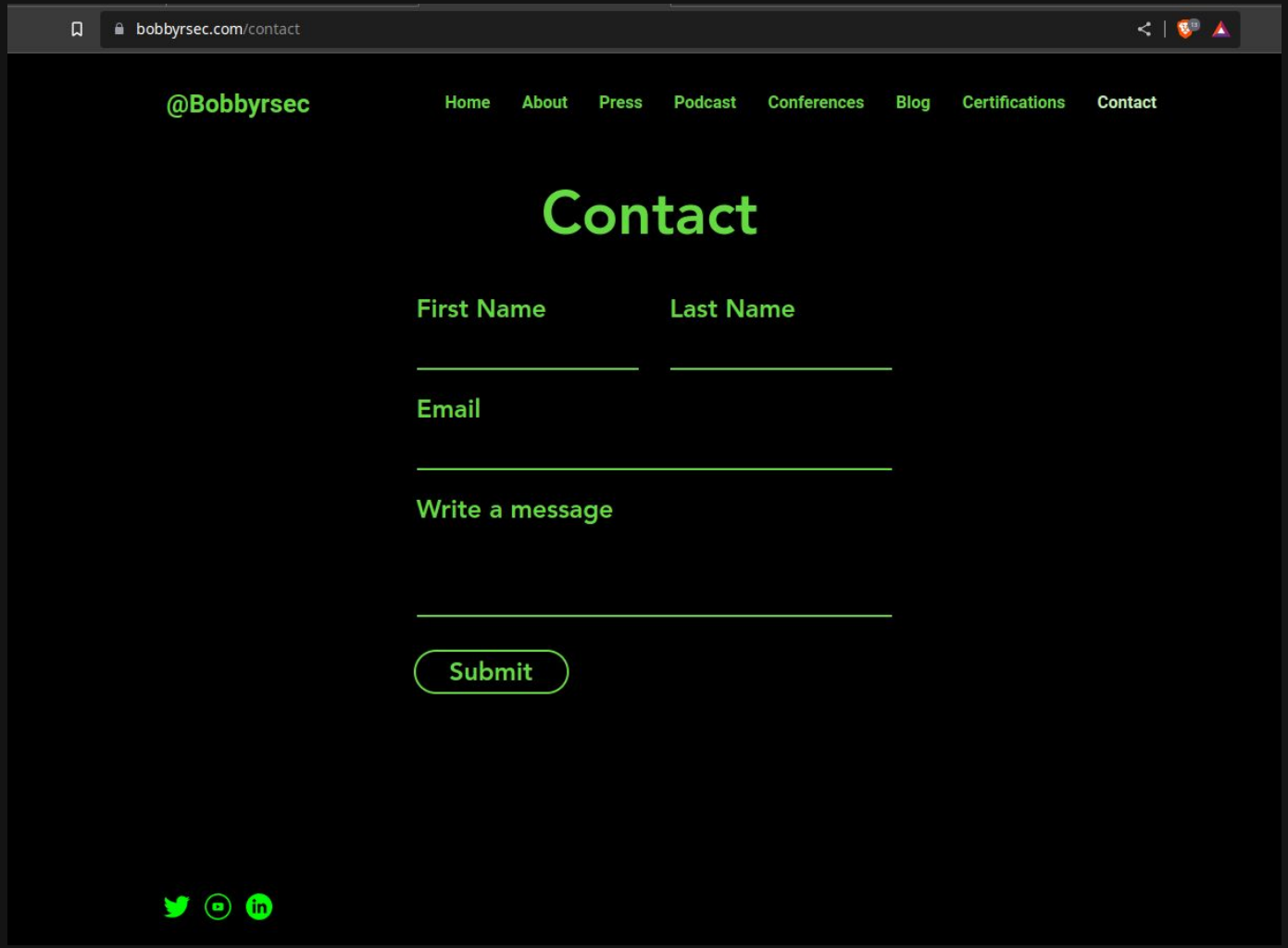
A CharlieCard being used at the State Street MBTA station in Boston. JOHN BLANDING/GLOBE STAFF

Bobbyr
Dec 8, 2022 · 18 min read · Listen

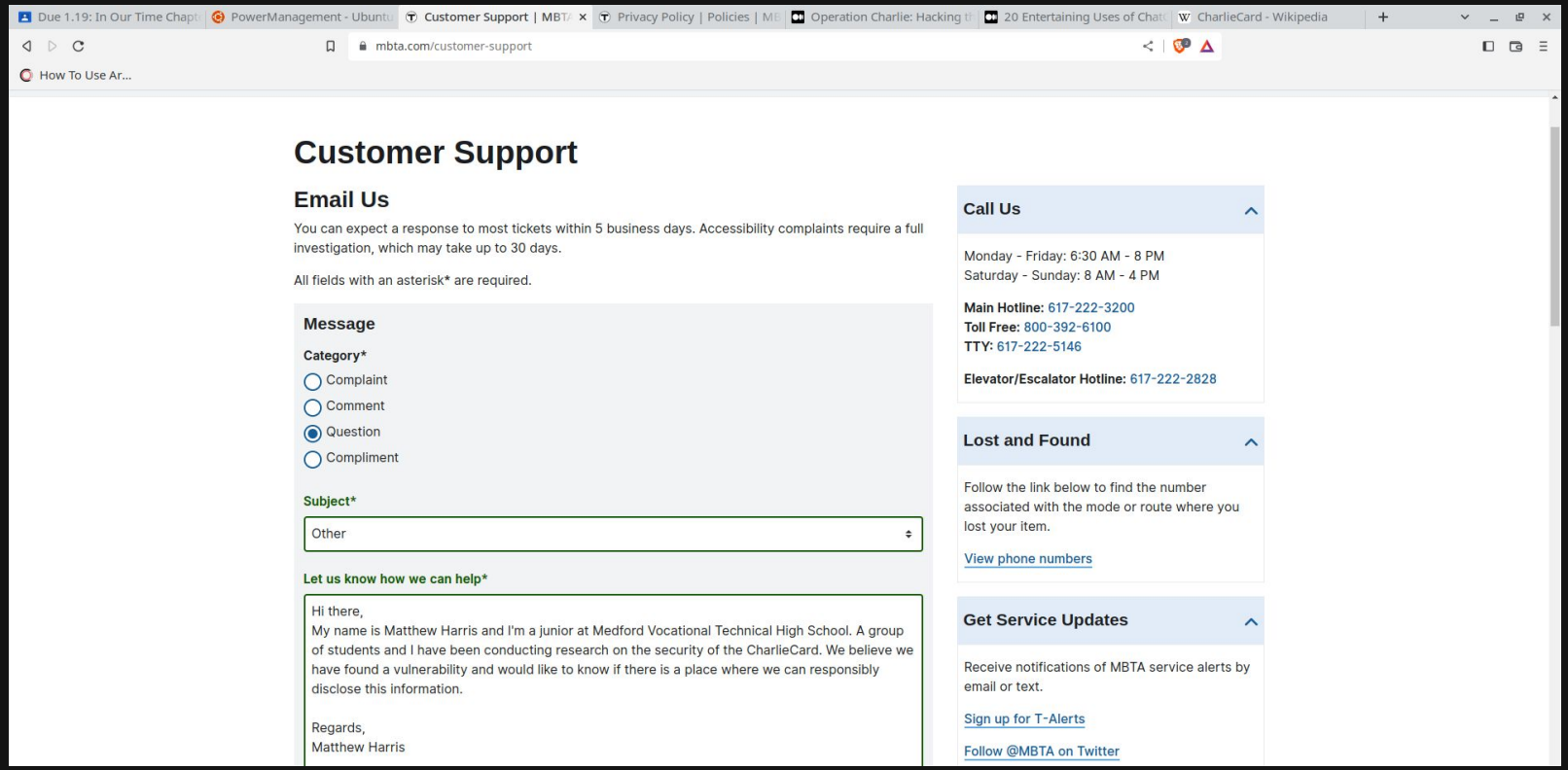
Operation Charlie: Hacking the MBTA CharlieCard from 2008 to Present

Photo by Garrett Quinn

December 8, 2022


bobbyrsec.com/contact
@Bobbyrsec
Home
About
Press
Podcast
Conferences
Blog
Certifications
Contact
Contact
First Name
Last Name
Email
Write a message
Submit

Reaching out to the T



January 2023

We Were Expecting This

And they responded!

Good afternoon, Matthew

Thank you for outreaching to us here at the MBTA and we do have a Vulnerability Disclosure Program (or VDP).

As part of the process, we will share terms for you to safely disclose, support us to understand the issue, safe-harbor provisions for you, and ultimately the timeline/terms so that you can publicly disclose or publish.

I'll work with our team to get this over to you in the next day.

In the interim, wanted to see if you are aware that we recently supported a similar case which was in the Boston Globe and local news regarding a CharlieCard vulnerabilities.

https://www.bostonglobe.com/2022/12/08/business/your-charliecard-can-be-hacked-by-an-android-phone-mbta-admits/

This was rather recent and not sure if you are aware. In any case, please let me know if you have any questions and in the interim, I'll get the terms to you.

Thank you again,

-Scott

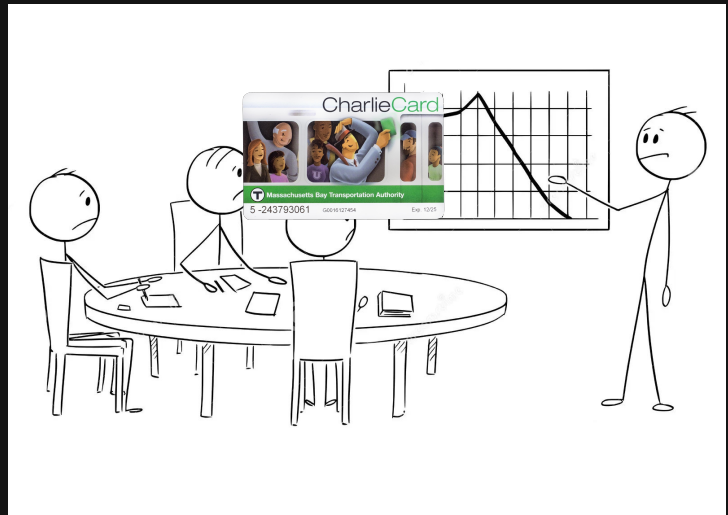
Scott Margolis | Chief Information Security Officer | | MBTA.com

Massachusetts Bay Transportation Authority

They Invited Us To Their Headquarters

The Meeting

- We rolled up
- It wasn't the address to train jail
- We sat down and gave them a more professional presentation
- They were very nice and talked to us about possible solutions



February 9, 2023

Post Meeting Fun

February 2023

We Were Very Slightly Suspicious

But Nobody Stopped Us…

But Nobody Stopped Us…

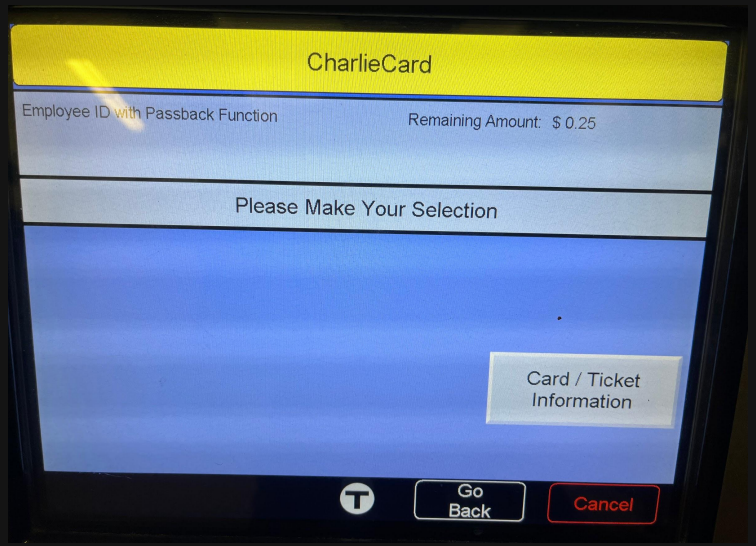
Until They Did

At 1AM

Take Two

February 2023

Post - Post Meeting Fun



March 2023

Demo Time

3 days ago

Recap and Takeaways

Recap

- We reverse engineered the MBTA's CharlieCards
- Gave ourselves free rides for life (or until they fix their system)
- Reported the vulnerability to the MBTA and met with their security team
- Created our own fare vending machine and an Android app

Takeaways If You’re A Hacker

- Free rides if you come to Boston
- Everything’s more fun with friends
- Reverse engineering strategies
- Reaching out to a government agency strategies

Things to Keep In Mind

- The same pitfalls should be avoided when the next system is rolled out
  - **Don’t store money on the card**
  - Make use of pre-existing secure systems such as Apple and Google Pay
- It's more secure to store cards, transactions, and users in a database
- Storing in a database also means that detecting vulnerabilities will be easier
- Databases also make studying analytics easier

- **Systems need to be maintained after they are made!**

Acknowledgments

Finn Sedan

- Pitched in money for our first NFC reader
- Drove us around
- Cool guy

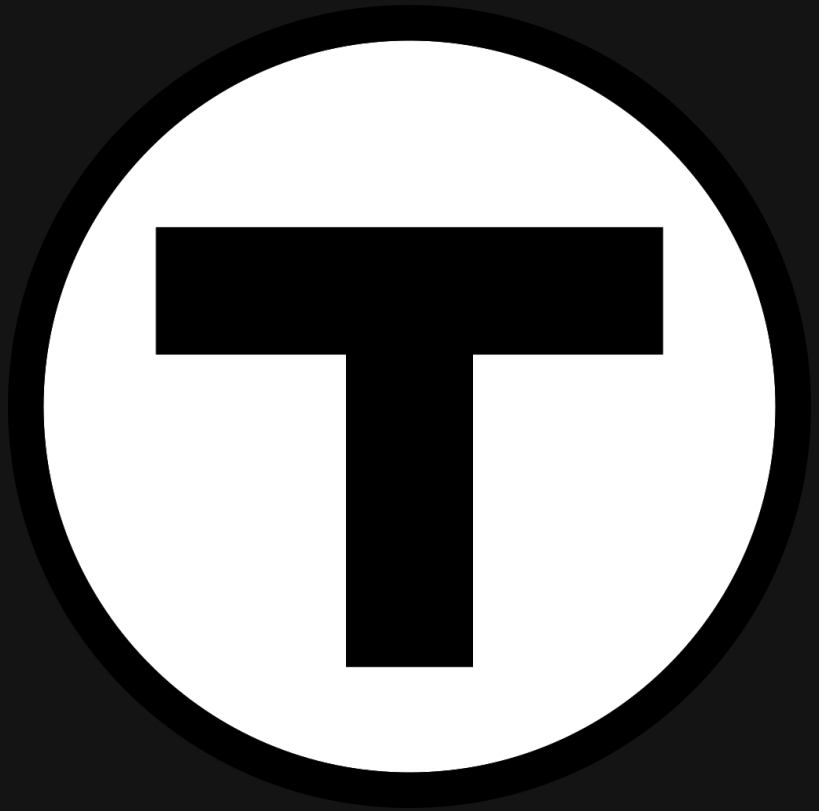
Bobby Rauch

- Helped us get in contact with the MBTA
- All around helpful guy

Mr. Christy

Mrs. Miller

MBTA



- Was very cooperative throughout the whole process
- Did not sue us
- Everyone was very nice to us
- And remember:

“The MBTA does not endorse or encourage this type of conduct. evading fares and/or hacking the MBTA fare system is illegal and the MBTA takes these matters seriously. Anyone caught engaging in this type of behavior will be referred to the appropriate law enforcement agency. The MBTA has implemented mitigation measures to address some of these concerns.”

If you wanna contact us

Matthew Harris - mharris06@protonmail.ch, mattyharris.net

Scott Campbell - joseph_scott_campbell@protonmail.com, josephscottcampbell.com

Zachary Bertocchi - zbertocchi@gmail.com, zackbertocchi.com

Noah Gibson - thebottemofthebarrel@gmail.com, noah.nopreserveroot.xyz

Image Credits

https://www.dreamstime.com/cartoon-businessman-presenting-bad-financial-results-business-work-meeting-cartoon-stick-figure-drawing-conceptual-image139220371

https://www.jing.fm/iclip/xRRii_detective-clipart-free-images-image-man-with-magnifying/

http://www.clipartbest.com/clipart-LTKdMyMGc

https://static.wixstatic.com/media/943b46_b7f1a659f1854cb3997d9f5ef20b6c2d~mv2.jpeg/v1/fill/w_238,h_238,al_c,q_80,usm_0.66_1.00_0.01,enc_auto/1672752960367.jpeg

https://dealer-communications.com/wp-content/uploads/2014/03/crossroads.jpg

https://static.wikia.nocookie.net/phineasandferb/images/f/f7/Doofenshmirtz_Evil_Incorporated.jpg/revision/latest?cb=20120523032047

https://www.cpusa.org/wp-content/uploads/2022/08/FBI-swat-team.jpg

https://www.bostonglobe.com/2022/12/08/business/your-charliecard-can-be-hacked-by-an-android-phone-mbta-admits/

https://medium.com/@bobbyrsec/operation-charlie-hacking-the-mbta-charliecard-from-2008-to-present-24ea9f0aaa38

Image Credits Part 2

https://static.wixstatic.com/media/943b46_b7f1a659f1854cb3997d9f5ef20b6c2d~mv2.jpeg

https://cdn.mbta.com/sites/default/files/fares/charliecard-tap-farebox.jpg

https://github.com/rust-lang/rust-artwork/blob/master/logo/rust-logo-512x512-blk.png

https://upload.wikimedia.org/wikipedia/commons/4/4c/Typescript_logo_2020.svg, ™/®Microsoft, Public domain, via Wikimedia Commons

https://upload.wikimedia.org/wikipedia/commons/7/76/Logo_SolidJS.svg, SolidJS, MIT <http://opensource.org/licenses/mit-license.php>, via Wikimedia Commons

Tauri Logo, https://tauri.app/about/trademark/

https://cdn.shopify.com/s/files/1/1061/1924/products/4_grande.png?v=1544200553

https://static.wikia.nocookie.net/monopoly/images/9/95/Chance_go_to_jail.jpg/revision/latest?cb=20121122151318

https://upload.wikimedia.org/wikipedia/en/9/9b/Get_out_of_jail_free.jpg?20170102004957

https://media.defense.gov/2017/May/08/2001743920/-1/-1/0/170506-Z-XH297-039C.JPG

https://assets.afcdn.com/story/20140922/497618_w980h638c1cx532cy248.jpg

Thanks for Watching!