Reverse Engineering a real-world RFID payment system

How the EasyCard allows you to print your own digital money

Harald Welte

hmw-consulting.de
gnumonks.org
gpl-violations.org
osmocom.org

27th CCC Congress, December 2010, Berlin/Germany

Outline

1. The EasyCard system
2. Analyzing the EasyCard
3. Tampering with the EasyCard

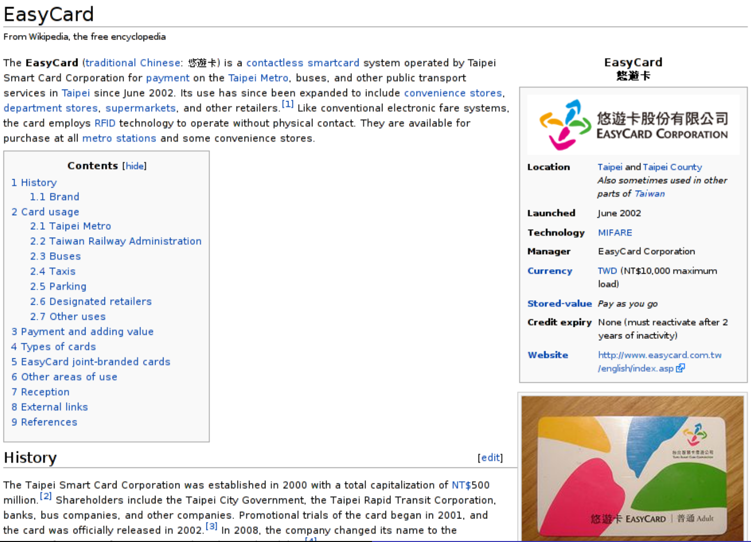
About the speaker

- Kernel / bootloader / driver / firmware development since 1999
- IT security expert, focus on network protocol security
- Core developer of Linux packet filter netfilter/iptables
- Board-level Electrical Engineering
- Always looking for interesting protocols (RFID, DECT, GSM)
- Open Source hardware/firmware/software for RFID: librfid, OpenPCD, OpenPICC

Travelling to Taipei

Starting from 2006, I was doing a lot of freelancing work for companies in Taiwan, resulting in numerous business trips to the capital Taipei. As soon as you use public transport, you notice they are using an RFID based system called EasyCard. This was just after having worked extensively on the **OpenPCD** RFID reader and **OpenPICC** RFID tag simulator.
However, work kept me too busy to ever have a look at the EasyCard until 2010.

What is this EasyCard?



EasyCard

From Wikipedia, the free encyclopedia

The **EasyCard** (traditional Chinese: 悠遊卡) is a contactless smartcard system operated by Taipei Smart Card Corporation for payment on the Taipei Metro, buses, and other public transport services in Taipei since June 2002. Its use has since been expanded to include convenience stores, department stores, supermarkets, and other retailers.[1] Like conventional electronic fare systems, the card employs RFID technology to operate without physical contact. They are available for purchase at all metro stations and some convenience stores.

EasyCard
悠遊卡

悠遊卡股份有限公司
EasyCard Corporation

Location	Taipei and Taipei County *Also sometimes used in other parts of Taiwan*
Launched	June 2002
Technology	MIFARE
Manager	EasyCard Corporation
Currency	TWD (NT$10,000 maximum load)
Stored-value	*Pay as you go*
Credit expiry	None (must reactivate after 2 years of inactivity)
Website	http://www.easycard.com.tw/english/index.asp

Contents [hide]

History

[edit]

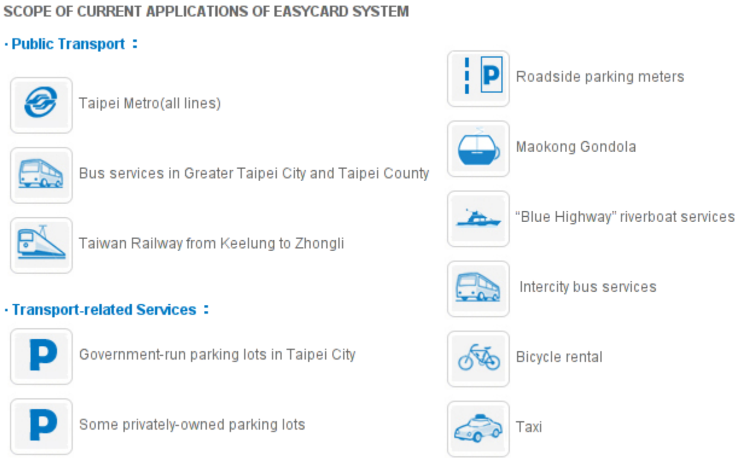
The Taipei Smart Card Corporation was established in 2000 with a total capitalization of NT$500 million.[2] Shareholders include the Taipei City Government, the Taipei Rapid Transit Corporation, banks, bus companies, and other companies. Promotional trials of the card began in 2001, and the card was officially released in 2002.[3] In 2008, the company changed its name to the

EasyCard

One of Asia's most popular electronic payment systems

- EasyCard is used in Taiwan, mostly in the capital Taipei
- Originally deployed in 2001
- More than 18 million issued cards
- Initially a payment system for public transport
  - Taipei metro (MRT)
  - Taipei public bus
- Similar to many other systems like Oystercard

EasyCard as payment in public transport



SCOPE OF CURRENT APPLICATIONS OF EASYCARD SYSTEM

· **Public Transport ：**

- Taipei Metro(all lines)
- Bus services in Greater Taipei City and Taipei County
- Taiwan Railway from Keelung to Zhongli
- Roadside parking meters
- Maokong Gondola
- "Blue Highway" riverboat services
- Intercity bus services
- Bicycle rental
- Taxi

· **Transport-related Services ：**

- Government-run parking lots in Taipei City
- Some privately-owned parking lots

EasyCard sale, recharge and refund

- Cards are purchased at vending machines located in every subway station
  - Price is 500 NTD: 400 NTD value, 100 NTD deposit
  - Payment is made in cash
  - Thus, no credit card / account number linking a person to a card
- Full refund of the account balance and the deposit can be made at a cashier
- Adding value to the card is made by the same machines that sell the cards

Threat analysis / Fraud potential

- It is publicly known that EasyCard uses NXP MiFARE
- MiFARE *Classic* has been broken in various ways before, ranging from eavesdropping attacks to card-only attacks.
- However, the card itself is only one element in the security chain
- EasyCard using MiFARE does not by itself mean that the EasyCard system is broken

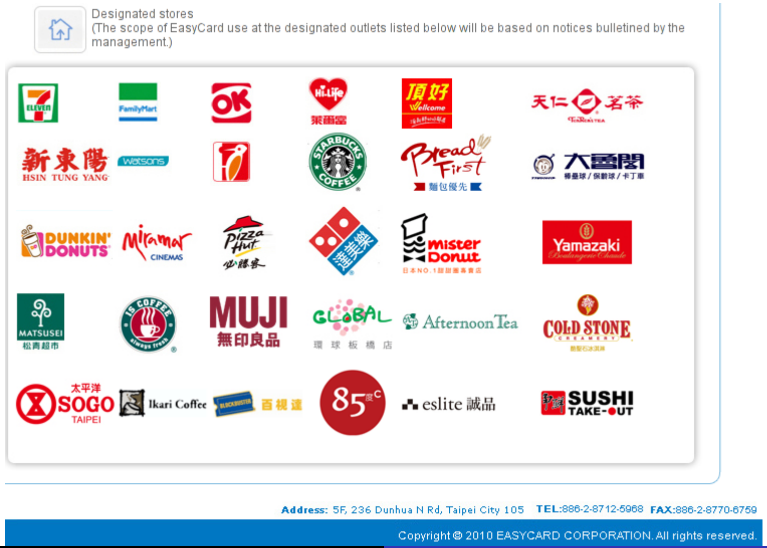
Online or Offline validation

- EasyCard could have been a relatively safe system, if
  - the value was not stored on the card but in the back-end
  - all transactions would inquire the back-end and not only the card
- I never really bothered to do much analysis, considering that all you could get is fraudulent free rides for public transport (which are cheap anyway)

EasyCard for payment in stores

- In 2009, the government creates laws for stored-value cards as means of payment
- In early 2010, use of the EasyCard is extended beyond public transport
    - you can store up to 10,000 NTD (240 EUR) on the card
    - the card is accepted at lots of stores (mostly big brands)
- The attack incentive is much higher: Not only free metro rides, but suddenly you can buy basically any goods available in the largest department stores

EasyCard as payment in stores



Designated stores
(The scope of EasyCard use at the designated outlets listed below will be based on notices bulletined by the management.)

Address: 5F, 236 Dunhua N Rd, Taipei City 105 TEL:886-2-8712-5968 FAX:886-2-8770-6759

What is MiFARE classic?

- A 13.56 MHz RFID card system based on ISO 14443 (1,2,3)
- 1024 or 4096 bits of storage, divided in sectors and blocks
- Uses proprietary 48bit cipher (CRYPTO1)
- Manufacturer and customers *really believed* in Security by obscurity ?!?
- Nobody should ever have used it for any application requiring security
- Weaknesses first published at 24C3 by Henryk Ploetz and Karsten Nohl

Analyzing the EasyCard

- First step: Verify it it indeed MIFARE classic
    - Can be done by applying ISO1443-1/2 air interface and ISO14443-3 anti-collision procedure and checking the result values
- Next step: Recovering the keys
    - many cards have one ore more sectors using the default manufacturer keys
    - if one sector key is known, breaking the other keys is fast/easy by means of a publicized existing attack
    - EasyCard uses custom keys for all sector, no success

Recovering the keys

- As all keys are unknown, the card-only *Dark Side* attack (Nicolas T. Courtios) was used
- Open Source `MFCUK` (MiFare Classic Universal toolKit) program implements the attack
- All hardware required is a RFID reader supported by libnfc (EUR 30)
- All A and B keys for all sectors have been recovered within 3 hours
  - Attack time could be much shorter if proxmark with very tight timing control was used

Extracting raw content

- Once the keys are known, the full data content of the card can be dumped
- Free Software `nfc-mfclassic` program (part of `libnfc`) was used
- All hardware required is a RFID reader supported by libnfc (EUR 30)

Re-engineering the data format

- The raw card content is not of much use unless it can be interpreted
- Individual transactions need to be made, raw card dumps acquired before/after each transaction
- Analysis of modifications caused by single transaction allow conclusions on data format
- Repeat this with transactions like
    - entering a metro station
    - leaving a metro station
    - recharging the card
    - purchasing something using the card

Sector 2: EasyCard balance

- MIFARE value blocks are intended for counters that can be incremented/decremented by different keys
- The actual counter value is stored three times (inverted/non-inverted) for safety
- EasyCard uses MIFARE value block in sector 2
- The value 1:1 represents the account balance of the card in NTD

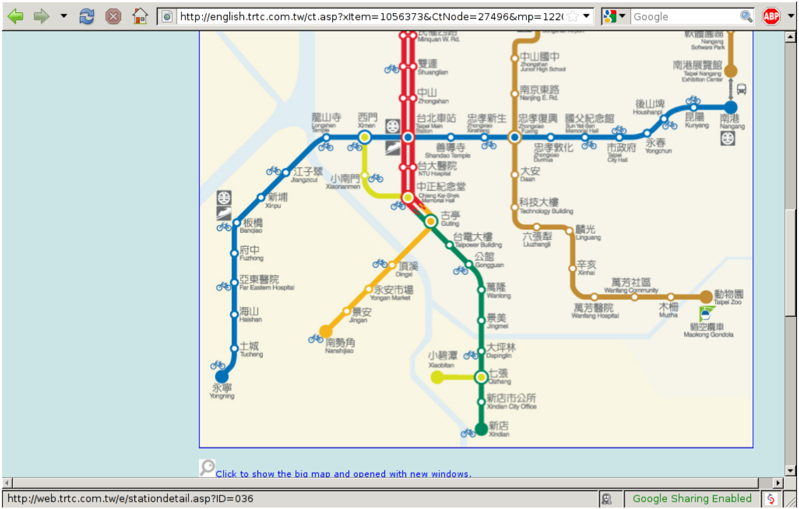
Sectors 3 through 5: Transaction Log

- Each 16-byte block in sectors 3 through 5 contains one transaction log record
- Each record contains
  - Transaction ID, Cost, Remaining Balance, MRT Station code, RFID reader ID
  - Transaction Type (Entering/leaving MRT, re-entering / connecting MRT, purchase, recharge
  - Timestamp is a 32bt unix time() format (seconds since January 1st 1970). However, it refers to CST instead of UTC

How to decode the MRT Station Code

- Transaction log record contains MRT station code
- How to know which station name corresponds to the numeric code?
  - Option A: visit each of them and take a EasyCard raw dump
  - Option B: visit the MRT homepage, point mouse at a specific station on the map and look at the URL: It contains the same ID!

EasyCard MRT station codes



http://english.trtc.com.tw/ct.asp?xItem=1056373&CtNode=27496&mp=122(

Google

Click to show the big map and opened with new windows.

http://web.trtc.com.tw/e/stationdetail.asp?ID=036

Google Sharing Enabled

Sector 7: Last MRT entry/exit record

- Block 2 (Offset 0x1e0) contains a record describing the last MRT station that was entered
  - Byte 4 contains the MRT station code
  - Bytes 9..12 contain a timestamp
- Block 1 (Offset 0xd0) contains a similar record describing the last MRT station that was left
- It is assumed that this information is used to compute the distance (and thus fee) to be paid for the current ride, as well as the discount that is made when switching from MRT to bus.

Sector 15: Maximum daily spending

- Block 2 (offset 0x3e0) contains a record keeping track of the amount of money spent on a single day
  - Bytes 0..10 are unknown (all zero)
  - Byte 11 contains the day of the month
  - Byte 12 contains an unknown value (0x3d on all tested cards)
  - Byte 13..14 contains the sum of all purchases on the indicated day
- This is used to impose a daily spending limit of NTD 3,000.

Tampering with the EasyCard

- After recovering keys + understanding the format, tampering with the card is easy
- Testing purchases with tampered card permits validation of the offline vs. online question
- Possible manipulations
  - Decreasing the value on the card
  - Increasing the value on the card
  - Bypassing the daily spending limit

Decreasing the value of the card

- Make a purchase in a store that accepts the EasyCard
- Find the transaction log entry and increase the cost of the purchase
- Decrement the value block storing the card balance by the same amount
    - Make sure you get the value block modifications right (inverted, non-inverted, backup copy)
- Alter the *amount spent per day* (Sector 15) to reflect increased amount

Decreasing the value of the card

- A card was manipulated accordingly
- The card behaved like expected, i.e.
  - it had less value remaining
  - it was still possible to use it in stores and public transport
  - the artificially removed money could not be spent
  - the card could still be re-charged at recharge machines, without ever recovering the artificially removed amount

Increasing the value of the card

- Make a purchase in a store that accepts the EasyCard
- Find the transaction log entry and **decrease** the cost of the purchase
- Increment the value block storing the card balance by the same amount
  - Make sure you get the value block modifications right (inverted, non-inverted, backup copy)
- Alter the *amount spent per day* (Sector 15) to reflect reduced amount

Increasing the value of the card

- A card was manipulated accordingly
- The card behaved like expected, i.e.
  - it had more value remaining
  - it was possible to use it in stores and public transport
  - the artificially removed money could all be spent (!)
  - the card could still be re-charged at recharge machines, without ever loosing the artificially added amount

NOTE: The artificially added money was immediately added by recharging the card at a recharge machine. The amount stored on the card has been reduced by the previously added amount. No fraud was committed!

Introducing `easytool`

- Information regarding the data format of the card implemented as C header file / structs
- C program `easytool` created to decode cards contents
- Later, code to decrement/increment amount was added
- Tool has not been released publicly
- Read-only version of the tool might be released soon

Summary

- Using MIFARE classic or any RFID system based on security by obscurity is irresponsible
- Extending a MIFARE classic based public transport payment system to general payment system in the year 2010 is nothing but ignorant, clueless and a sign of gross negligence
- Government regulartors should mandate the use of publicly and independently audited and reviewed security technology. Security by obscurity is not an answer to any problem.

Thanks

I would like to express my thanks to

Brita and Milosch Meriac	for OpenPCD and OpenPICC
Henryk Ploetz, Karsten Nohl, starbug	for their work on CRYPTO1
Jonathan Westhues	for his work on Proxmark
Nethemba	for implementing the nested key attack in MFOC
Roel Verdult	for libnfc
Nicolas T. Courtois	for his *darkside* paper
Andrei Costin	for his MFCUK implementation of the *darkside* paper