Some slides contains animated elements that are not available in the PDF format

Breaking the HID iCLASS Standard

By Michael Cahyadi

Originally published in 2019
Translated from indonesian to English

Everything done in this presentation is done for
educational purposes only

Information regarding **classified proprietary information owned by HID** will be **redacted**

RFID Security (1st Edition)
ISBN 1-59749-047-4
Syngress, 2005

RFID is

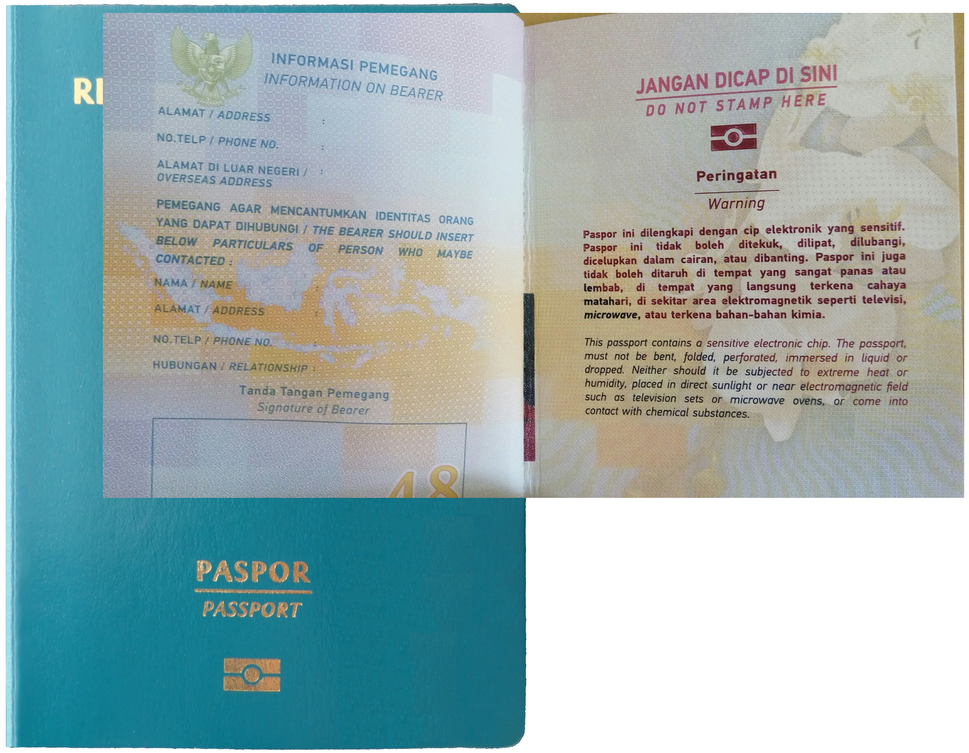
RFID Security (1st Edition)
ISBN 1-59749-047-4
Syngress, 2005

Radio Frequency IDentification

Radio Frequency IDentification is used in

RFID Security (1st Edition)
ISBN 1-59749-047-4
Syngress, 2005

E-Pasports



Radio Frequency IDentification is used in

RFID Security (1st Edition)
ISBN 1-59749-047-4
Syngress, 2005

E-Pasports

Payment Cards

It comes in different flavors

Technology Card Guide
HIDGlobal, 20 May 2015

Nama	Frequency	Reach	ISO Terminology
Low Frequency (LF)	125 kHz	<45cm	Proximity Cards (ISO 7810)
High Frequency (HF)	13.56 MHz	45-100cm	Contactless Cards (ISO 24727)

But we are focusing on one

HID Proximity Brochure
HIDGlobal, 02 Oct 2018

iCLASS Product Brochure
HIDGlobal, 2007

Smart Contactless Card

- Uses 13.56 MHz RFID Frequencies
- Encrypted using DES/Triple-DES (Digital Encryption Standard)
- the CSN is protected

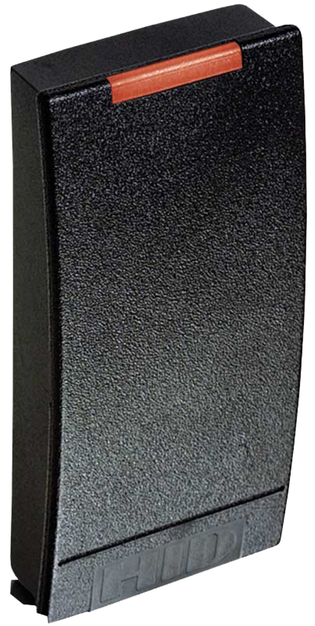
Before iCLASS there was ProxCard that transmits it's UID in plaintext

iCLASS Product Brochure
HIDGlobal, 2007

Proximity Card (ISO 7810)

6d 61 69 6b 20 73 61 79 61 6e 67 20 73 61 6e 73 61 6e

Reader



*this is **not a real UID**

So iCLASS uses...

Cryptography Engineering
ISBN 978-04704744242
Bruce Scheneier, 2010

Encryption

Cryptography Engineering
ISBN 978-04704744242
Bruce Scheneier, 2010

HID iCLASS uses the Hash0 algorithm to create all signatures

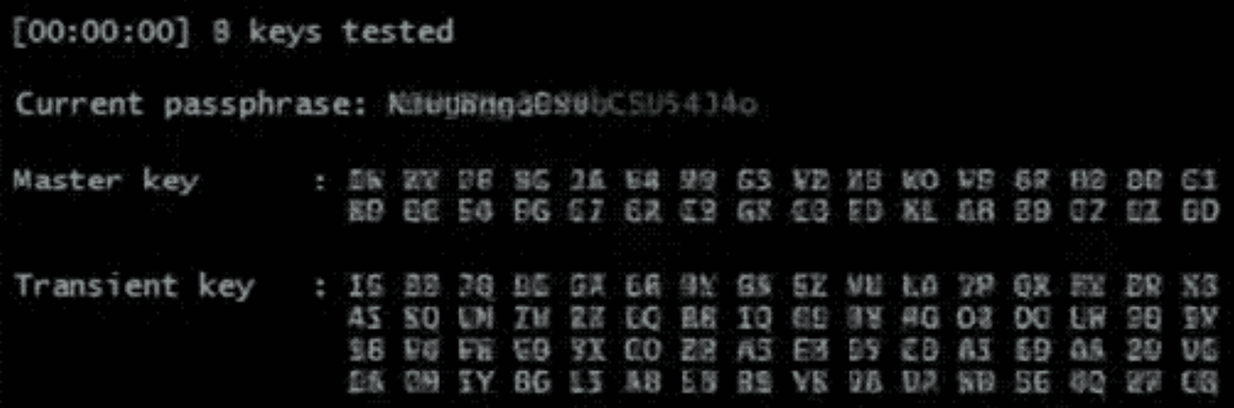
Cryptography Engineering
ISBN 978-04704744242
Bruce Scheneier, 2010

Using the signature in the reader, the device can iterate unique keys for every card

5176 0400 0002 5018

Cryptography Engineering
ISBN 978-04704744242
Bruce Scheneier, 2010

Using the We could attempt to bruteforce the combination but this method requires significant computing power if done without prior information gathering and supplemental information in the reader



It also has a unique quirk

iCLASS Product Brochure
HIDGlobal, 2007

Smart Contactless Card

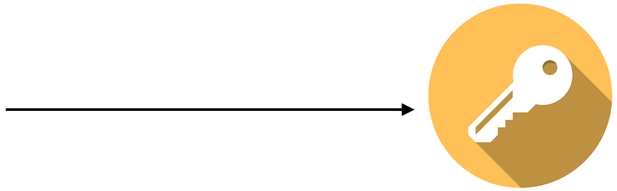
Reader checks if the key in the card and the key in the reader are the same

Reader

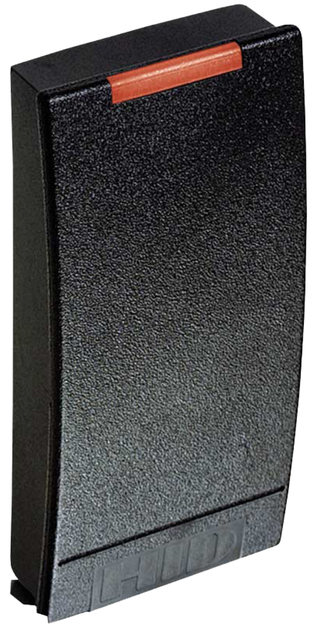
If the keys are the same

iCLASS Product Brochure
HIDGlobal, 2007

Smart Contactless Card



Reader



The CSN is transmitted and authenticated

iCLASS Product Brochure
HIDGlobal, 2007

Smart Contactless Card

Reader

So there are layers to this...

Level	Layers of Security
Lvl0	CSN
Lvl1	phdr + rf signal (IDENTIFY)
Lvl1	DES Signature (Q) #N Data
Lvl2	INCrypt32 #0 data
Lvl2	Hash0 #0 data
Lvl3	TDES Key *(R,S)* #N data

Smart Contactless Card

So we just have to

Smart Contactless Card

1. Extract the CSN
2. Extract the TDES Key
3. Duplicate the card

Smart Contactless Card

First Step
Extracting the CSN

On the left is a device called an OMNIKEY, it's what HID calls a Secure Access Module that can read and write the contents of an HID iCLASS card. This is commonly used by technicians for the initial programming of the card.

OMNIKEY 5321 Product Brochure
HIDGlobal, 2007

Omnikey

$$\xleftarrow{id, c_C}$$

$$\xrightarrow{n_R, a_R}$$

$$\xleftarrow{a_C}$$

Smart Proximity Contactless Card

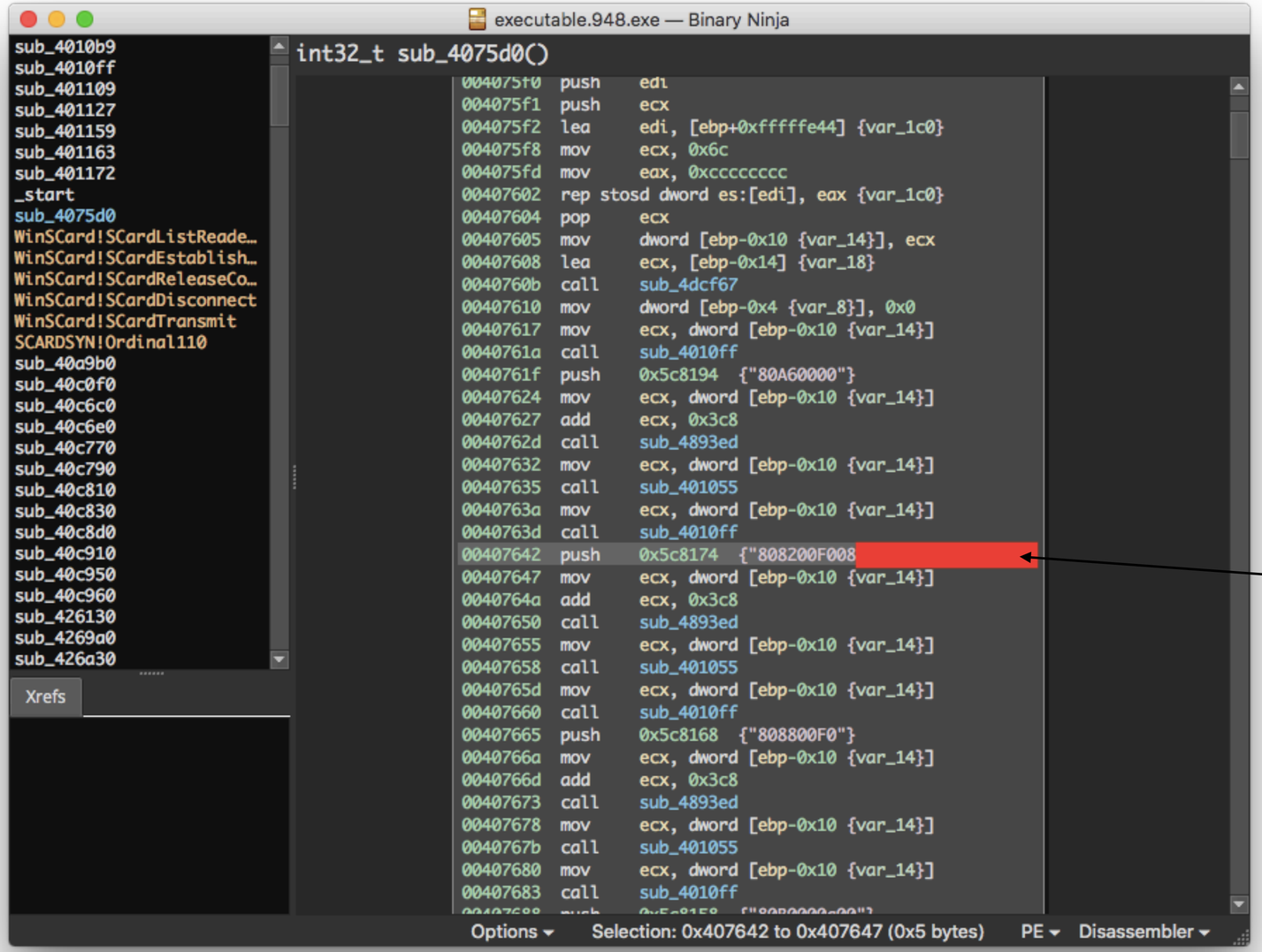
OMNIKEY 5321 Product Brochure
HIDGlobal, 2007

Reader/Writer Multikey

- It can do read-write operations
- It's contents are encrypted with TDES too
- The USB connection to a PC is secured using somesort of **Secure Mode**

Reader/Writer Multikey

In the **Secure Mode**, there is a security flaw in an old driver that accidentally gives the **user root access of the device**



Master Authentication Key

By disassembling the original OMNIKEY firmware, we can find the default master authentication key that is contained in every single OMNIKEY device

```
MINGW32:~/iclassified
Administrator@user-09118738a3 ~/iclassified
$ iclass.exe read
fcee5601f7ff12e0
12ffffff7f1fff3c
feffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
030303030003e017
2ed49bc872bed6d4
2ad4c8211f996871
2ad4c8211f996871
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff
ffffffffffffffff

Administrator@user-09118738a3 ~/iclassified
$ iclass.exe write 7 4141414141414141
SUCCESS

Administrator@user-09118738a3 ~/iclassified
$ iclass read 7
4141414141414141

Administrator@user-09118738a3 ~/iclassified
$ _
```

CSN
(Card Serial Number)

Reader Parity Data

Reader/Writer Multikey

We can use that masterkey to create a program that can read the card and acquire the CSN from the program header, without the use of HID's proprietary card decoding software

Time to go deeper...

Level	Levels of Security
~~Lvl0~~	CSN
Lvl1	phdr + rf signal (IDENTIFY)
Lvl1	DES Signature (Q) #N Data
Lvl2	INCrypt32 #0 data
Lvl2	Hash0 #0 data
Lvl3	TDES Key *(R,S)* #N data

Smart Contactless Card

Second Step
Extract the TDES Key

Scanner

USB Cable

Rewriteable Test Cards

For this step we will be using a device called Proxmark that was built by Johnathan Westhues. This device contains a low frequency and high frequency band antenna that connects to an onboard chip to extract keys from the card and reader. The blueprints for a Proxmark device is available online, however we have decided to add some of our own modifications such as lowering the memory capacity to lower the production cost and speed up production.

In this step we use the Proxmark device to fool the reader into thinking that the device is a genuine HID card. This is done to take the frequency response that is created by the reader.

iCLASS Key Extraction
Jonathan Westhues, 2010

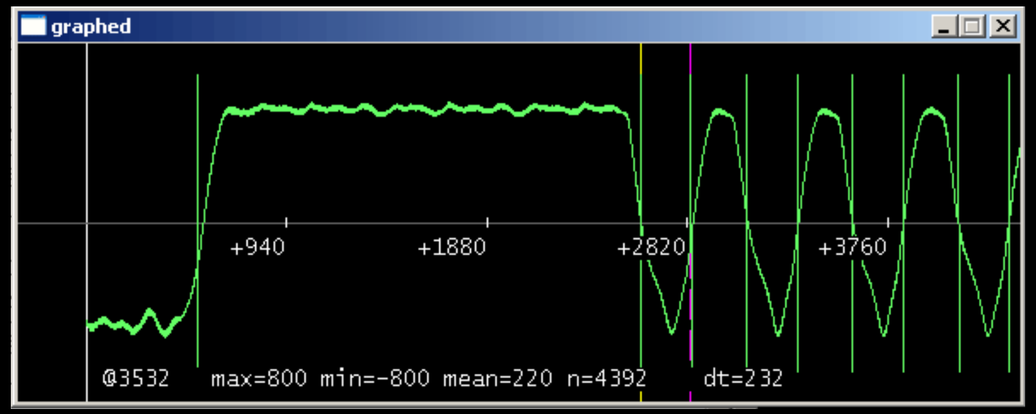
Reader

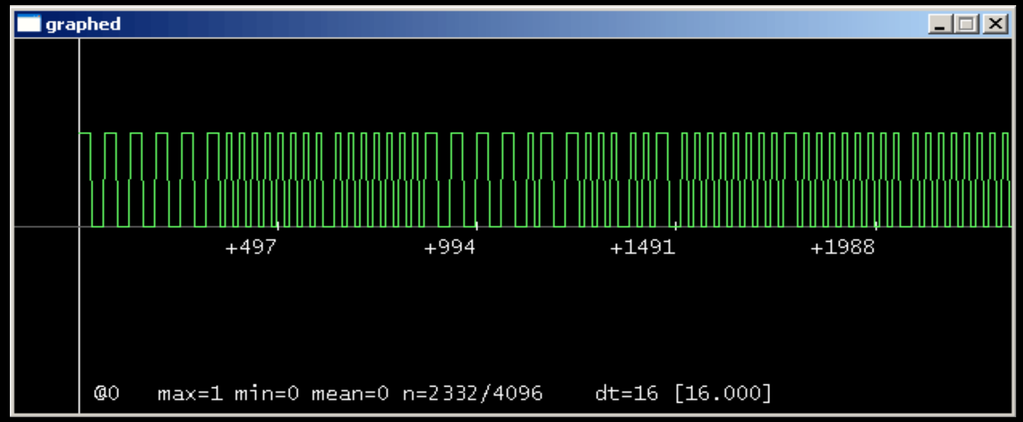
IDENTIFY

Proxmark 3

Key DES

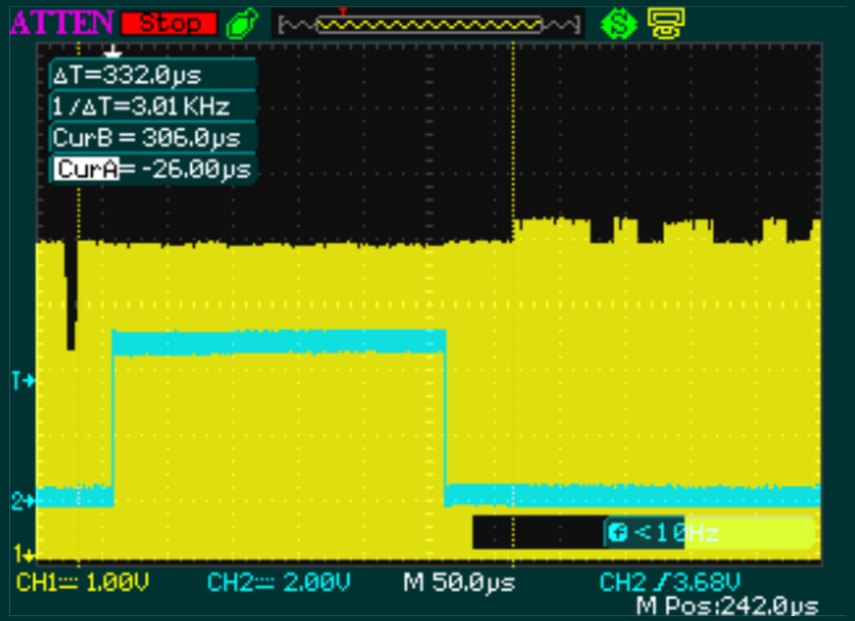
We need to match the proxmark radio frequency with the frequency that is expected of a genuine HID iCLASS card



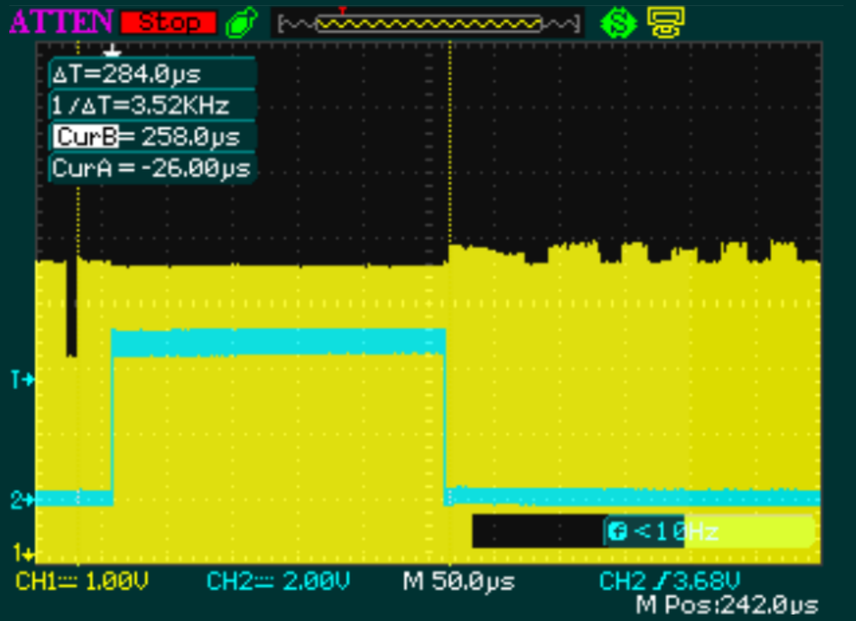


Spectogram results from Martin Holst Swende

Response of an **"IDENTIFY"** signal from a genuine reader



Response of an **"IDENTIFY"** thats **replicated**



Next on the chopping block

Level	Levels of Security
Lvl0	CSN
Lvl1	phdr + rf signal (IDENTIFY)
Lvl2	DES Signature (Q) #N Data
Lvl2	INCrypt32 #0 data
Lvl3	Hash0 #0 data
Lvl3	TDES Key *(R,S)* #N data

Exposing iCLASS Key Diversification
Radboud University, 2010

We will attempt to break the implementation of the algorithm used by HID. In the iCLASS system there is a formula to calculate the keys and in the keys there are several parameters.

Publicly shared :

P, u, b, G (parameters)

Q = Public key
e = Hash of data
R = Signature by scalable multiplication
S = Signature by normal numbers

Privately stored :

M = random number
Kd = debit key

Level	Levels of Security
Lvl0	CSN
Lvl1	phdr + rf signal (IDENTIFY)
Lvl1	DES Signature *(R,S)* #N Data
Lvl2	INCrypt32 #0 data
Lvl2	Hash0 #0 data
Lvl3	TDES Key *(Kd)* #N data

Signature is a pair of variables *R, S* thats programmed as :

$$R = (mG)_x$$

$$S = \frac{e + kR}{m}.$$

The *m* varibale needs to be random for the encryption system to work. If a *signature* uses the same *m* value, a *user* can calculate *k*.

There are two S Variables (one for the reader and the other for the card) so there are S_1 and S_2

$$R = (mG)_x \qquad R = (mG)_x$$

$$S_1 = \frac{e_1 + kR}{m} \qquad S_2 = \frac{e_2 + kR}{m}$$

$$R = (mG)_x \qquad R = (mG)_x$$

$$S_1 = \frac{e_1 + kR}{m} \qquad S_2 = \frac{e_2 + kR}{m}$$

If m has the same value in the two signatures, then R will be the same

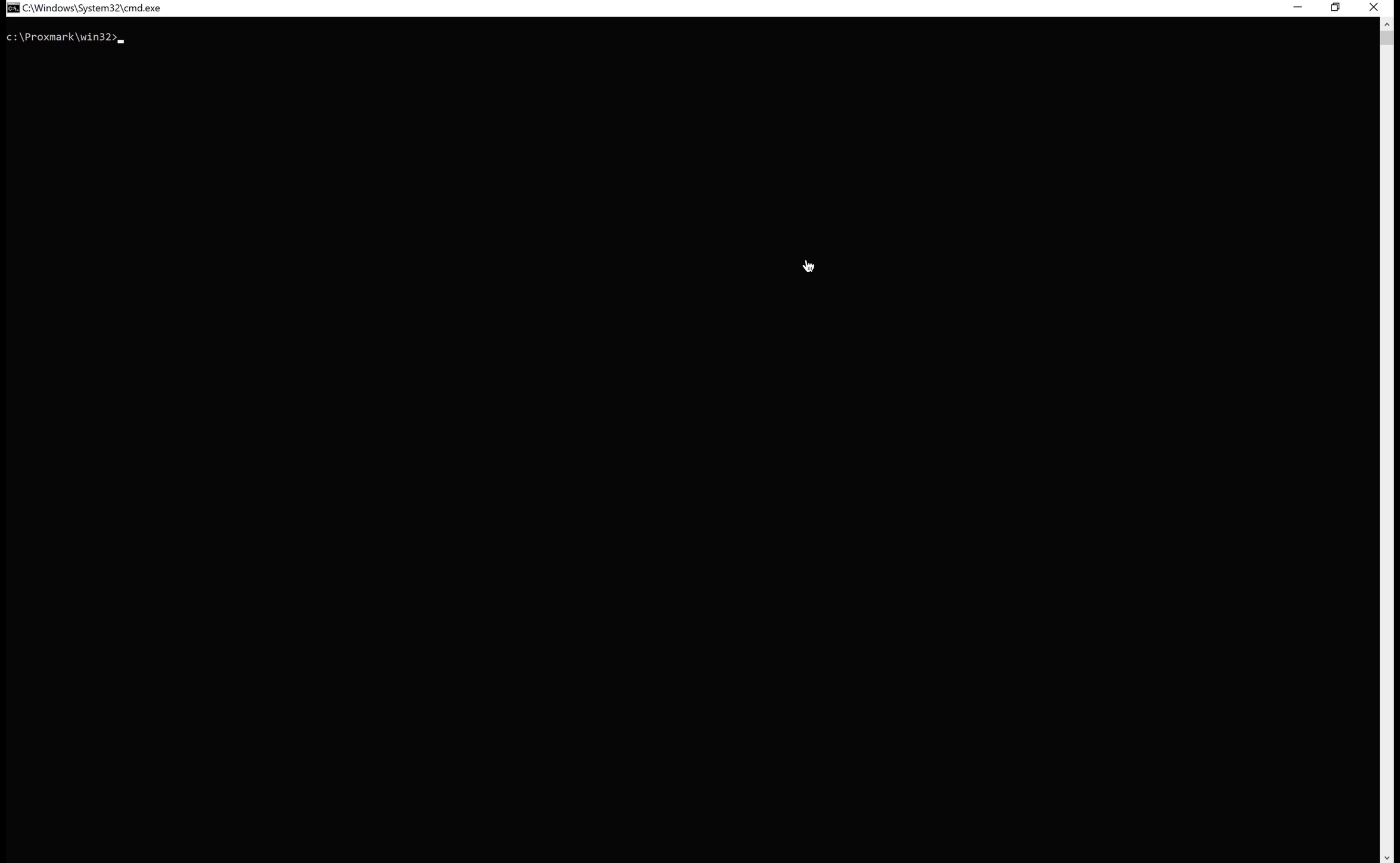
$$S_1 - S_2 = \frac{e_1 - e_2}{m}$$

$$m = \frac{e_1 - e_2}{S_1 - S_2}$$

$$k = \frac{mS_i - e_i}{R} \qquad \left[= \frac{e_1 S_2 - e_2 S_1}{R(S_1 - S_2)} \right].$$

With this we not only took the DES signature but we also obtained the TDES key which is the last level of security in the iCLASS ecosystem.

Level	Levels of Security
L0	CSN
L1	phdr + rf signal (IDENTIFY)
L2	DES Signature (Q) #N Data
L2	INCrypt32 #0 data
L3	Hash0 #0 data
L3	TDES Key *(R,S)* #N data


C:\Windows\System32\cmd.exe
c:\Proxmark\win32>

Smart Contactless Card

Step 3
Replication

```
pm3 --> hf tune
Measuring antenna characteristics, please wait...
#db# DownloadFPGA(len: 42096)
....
# LF antenna: 35.20 V @   125.00 kHz
# LF antenna: 33.83 V @   135.00 kHz
# LF optimal: 35.71 V @   127.66 kHz
# HF antenna: 30.77 V @    13.56 MHz
Displaying LF tuning graph, Divisor 89 is 134khz, 95 is 125khz

pm3 --> hf iclass managekeys n 0 k aea684a6dab23278
#db# SUCCESS
```

#db# going into sicko mode, 8 CSNS sent

```
pm3 --> hf iclass eload iclass_tagdump-aa16230f8ff12f1.bin
Sent 42096 bytes of data to device emulator memory

pm3 --> hf iclass sim 0 e53d1d0efeff12e0
#db# Simulating CSN e53d1d0efeff12e0
```

Authors Suppressed Due to Excessive Length

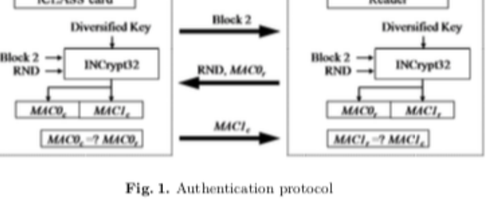


Fig. 1. Authentication protocol

te MAC. At this point, the card can compute an 8-byte MAC in the ay. If $MAC0_r$ is correct, the card will answer the other 4-byte signature $_t$) that enables the reader to authenticate the card. Therefore, the au- tion protocol needs to perform INCrypt32 with 12-byte input and 8-byte data.

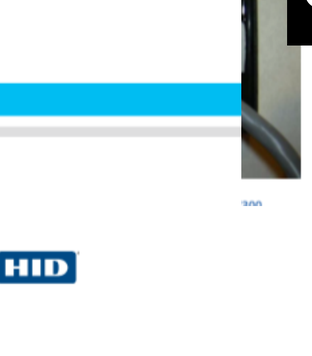
Protocol If the authentication protocol succeeds, the reader is able to ta blocks without an additional authentication procedure. However, to n 8-byte data in a data block, the reader needs to perform INCrypt32 me. The write protocol is described in Fig. 2.

In Circuit Serial Programming (ICSP) Command		
ICSP Data/ PIC Instr.	18F452 PIC Assembly Code	Comment
0x0E00	MOVLW, 0	Set Upper byte of Inde
0x6EEA	MOVWF, PSR0H	
0x0E00	MOVLW, 0	Set Lower byte of Inde
0x6EE9	MOVWF, PSR0L	
0x50EE	MOVF, POSTINC0	Read File Register & I
0x6EF5	MOVWF, TABLAT	Move Reg data to ICSP
Reg Data	N/A	Send data byte read to

Capture Circuit Implementation

ire required to extract the iClass register information is fai
rcuit is comprised of a generic 8-bit microcontroller which
an RS-232 transceiver, a couple of push buttons and a cou
ICSP interface and a PC serial COM port. The microcontro
he ICSP interface. The serial EEPROM is used to store the
d across the ICSP interface. The RS-232 transceiver is used
mp the captured data at a later time. The RS-232 transceiv
9Vdc (min) needed to force the PIC ICSP interface into a d

nload, the capture circuit receives its operating power dir
the readers ICSP connector. In addition, an onboard 5Vd
9Vdc battery which is used to power the circuit during the
a PC.



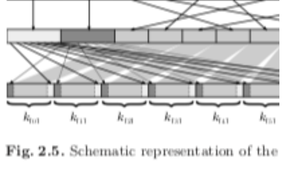


Fig. 2.5. Schematic representation of the

Remark 3. The DES implementation used in iClas NIST standard [12] in the way of representing keys. DES key is of the form $\langle k_0 \dots k_6 p_0, k_7 \dots k_{13} p_1, \dots,$ are the actual key bits and $p_0 \dots p_7$ are parity bits. I is of the form $\langle k_0 \dots k_{55} p_0 \dots p_7 \rangle$.

The following sequence of definitions describe the fu function is included here for the sake of complete construction are not necessary to understand the att and Section 3.3.

Definition 8. *Let the function check:* $(\mathbb{F}_2^6)^8 \to (\mathbb{F}_2^6$

$$check(z_{[0]} \dots z_{[7]}) = ck(3, 2, z_{[0]} \dots z_{[3]}) \cdot c$$

where $ck: \mathbb{N} \times \mathbb{N} \times (\mathbb{F}_2^6)^4 \to (\mathbb{F}_2^6)^4$ *is defined as*

$$ck(1, -1, z_{[0]} \dots z_{[3]}) = z_{[0]} \dots z_{[3]}$$

$$z_{[0]} \dots z_{[3]})$$

$$z_{[0]} \dots z_{[k]} \leftarrow$$

$$z_{[0]} \dots z_{[3]}\rangle,$$

$$ute: \mathbb{F}_2^n \times ($$

$$permute(p_0 \dots p_n, z, l, r) = \begin{cases} (z_{[l]} + 1) \cdot permute(p_0 \dots \\ z_{[r]} \cdot permute(p_0 \dots p_{n-} \end{cases}$$

Definition 10. *Define the bitstring* $\pi \in (\mathbb{F}_2^8)^{35}$ *in h*

$$\pi = 0x0F171B1D1E272B2D2E33353$$
$$4D4E535556595A5C636566696$$

Each byte

2 Key Diversification

2.1 Construction

For diversification, the recommended way by NXP is to use the CMAC construction of an amount of data using a master key. See [CMAC].

The pre-requisite is that there is enough input "diversification data" in order to make it a MAC. A MAC is used rather than encryption to make it a one-way function.

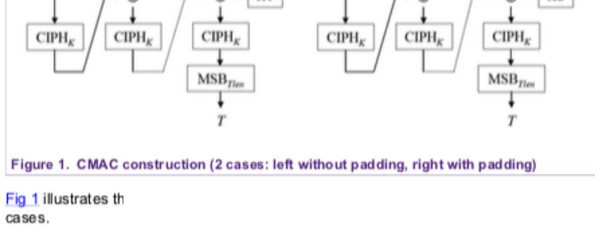


Figure 1. CMAC construction (2 cases: left without padding, right with padding)

Fig 1 illustrates th
cases.

According to [CM
is modified before
(denoted K1 or K2
the choice of whic
contains padding

These computatio
context of the key
because the padd
computations can
key to be diversifi
signals to the CM

If the keys are to

THE DARK SIDE OF SECURIT
and Cloning MiFare Classic Rail and Buildi

Nicolas T. Courtois
University College London, Computer Science, Gower

iCLASS Compatibility Chart

	Card Compatibility	R10, R30, R40, RK40	RW300, RW400, RWK400, RWKL550, RWKB575	
ISO 15693	Card CSN Read	2K & 16K	2K & 16K	Infi Phi
	Card Read	2K & 16K	2K & 16K	
	Card Write	NONE	2K & 16K	
ISO 14443A	Card CSN Read	Philips MIFARE, UltraLight, DESFire™	Philips MIFARE, UltraLight, DESFire™	
	Card Read	NONE	NONE	
	Card Write	NONE	NONE	

HID

How a Credential is 'Read'

Smart Card Door Access Employing Mutual Authentication

Credential and reader establish they are part of a trusted population (not forged, cloned, or spoofed)

If they establish trust, the process continues as normal.

comes from a modulo operation. Here odulo 62, which is 111110 in binary. x78:

$$(z_6 \bmod 62) + 2 = .111110.$$
$$(z_6' \bmod 62) + 2 = .000010. \oplus$$
$$01111000 = 0x78$$

utput variations invoked by bitflips in ion $z_6 + 1 = z_7$. The corresponding $k_1^{\oplus}$ $k_{1[1..6]} = 1$ when the relation holds and 2) + 2 when it does not hold. Example

$$\text{mod } 62) + 2 = .100110.$$
$$\text{mod } 62) + 1 = s_7) = .000001. \oplus$$
$$01001110 = 0x4e$$

tion for $k_{1[1..6]}$ is:

$$(z_6 \bmod 62) + 1 = (z_7 \bmod 63);$$
$$162) + 2, \qquad otherwise.$$

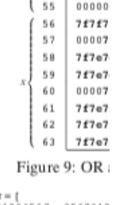


Figure 9: OR

$\pi = [$
01234567, 35670124,
34670125, 01352467,

Keywords: Access control, RFID, contactless smart cards, MiFare Cla secrets, secure hardware devices, reverse-engineering, elec backdoors, critical application development management,

Abstract: MiFare Classic is the most popular contactless smart card wide. At Esorics 2008 Dutch researchers showed that th little as 0.1 seconds if the attacker can access or eavesdrop We discovered that a MiFare classic card can be cloned i the attacker only needs to be in the proximity of the card fo of identity through pass cloning feasible at any moment an sitting next to the victim on a train or on a plane is now be also (independently from us) discovered this vulnerability queries to the card and does not require any precomputati or clones of MiFare Classic are even weaker, and can be cl The main security vulnerability that we need to address wi phy, RFID protocols and software vulnerabilities. It is a s economy is vulnerable to sophisticated forms of electronic oper can intentionally (or not), but quite easily in fact, c and financial institutions worldwide.

about the keys

Michael Cahyadi <maikxchd@gmail.com>
to Matthew

Ok, I have hash1 down. No problem there.

but according to above z0 = e9 11 52 2b 3f dd 8e e8

but when i try z0=des_ecb(key, ~key) = f2 e7 33 d7 35 fe ae 2c

and I thought Ksel(i)=h(hash1(id)(i)) (where h = hash2(Kcus))
so I thought 00 = 8a, 1c = 0d ....

Is it possible to have another hash2 example, to figure out where Ive obviously gone wron

Thanks

Reply Forward

z0 was suppo
permutated k

but its not respondi
somehow

try turning the omnikey o

then off again

then on again

ok th

re can i go to bed now?

olio

iceman 2019-04-18 03:22:14

Administrator

Registered: 2013-04-25
Posts: 5,343
Website

Online

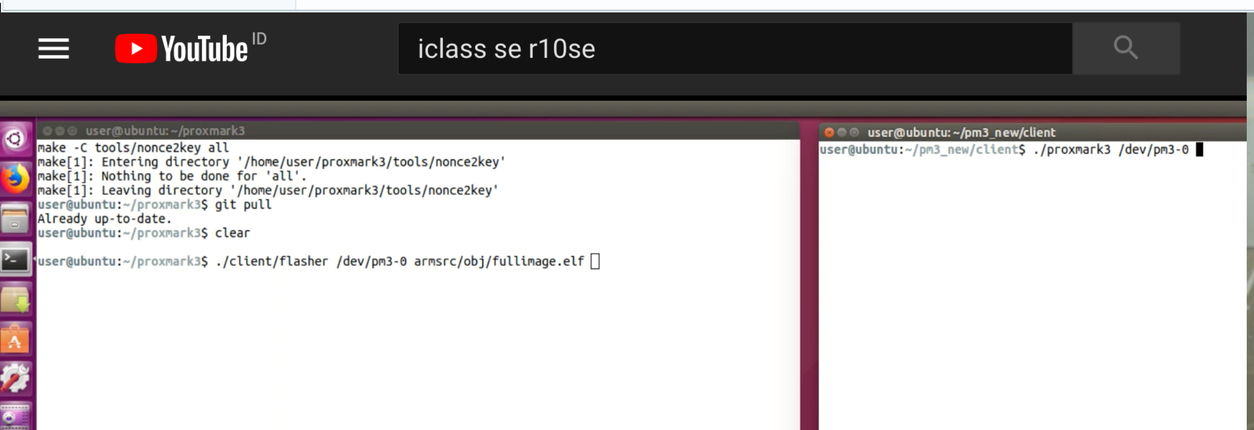
Usually I make sure the t5577 is writeable and "detected" before I
"detected", but I write the block 1-3 and write config block at the
commands and output. Dunno why you try the pwd in the writes.

When verifying, you need detect, then dump it again.

Which kind of tag is this? Never seen this config block before?

If you feel the love, https://www.patreon.com/iceman1001

modhex(hkhehghthbhudcfcdchkigiehgduiehg)



special thanks for

- **Jonathan Westhues (Oregon, USA)**
Provided blueprints for the Proxmark 3 device

- **Flavio D. Garcia from Radboud University Nijmegen (Nijmegen, Netherlands)**
Provided the key diversification function for *HID iCLASS* cards

- **Christian Herman from IceSQL AB (Gotenburg, Sweden)**
Created a special firmware for the OMNIKEY and documented the schematics of the iCLASS reader

- **ELECHOUSE (Shenzen, China)**
For printing the Proxmark 3 Device

- **Kevin Chung dari New York University (New York, USA)**
Developed additional drivers for the OMNIKEY 5321 model and reverse engineered the *HID iCLASS* masterkeys

- **Fanny Soedjahdjo dari Philip Pocock Catholic Secondary School (Ontario, Canada)**
Provided additional material translation from english to indonesian
(this presentation was originally created in Indonesian)