EMV

Erik Poll

Digital Security

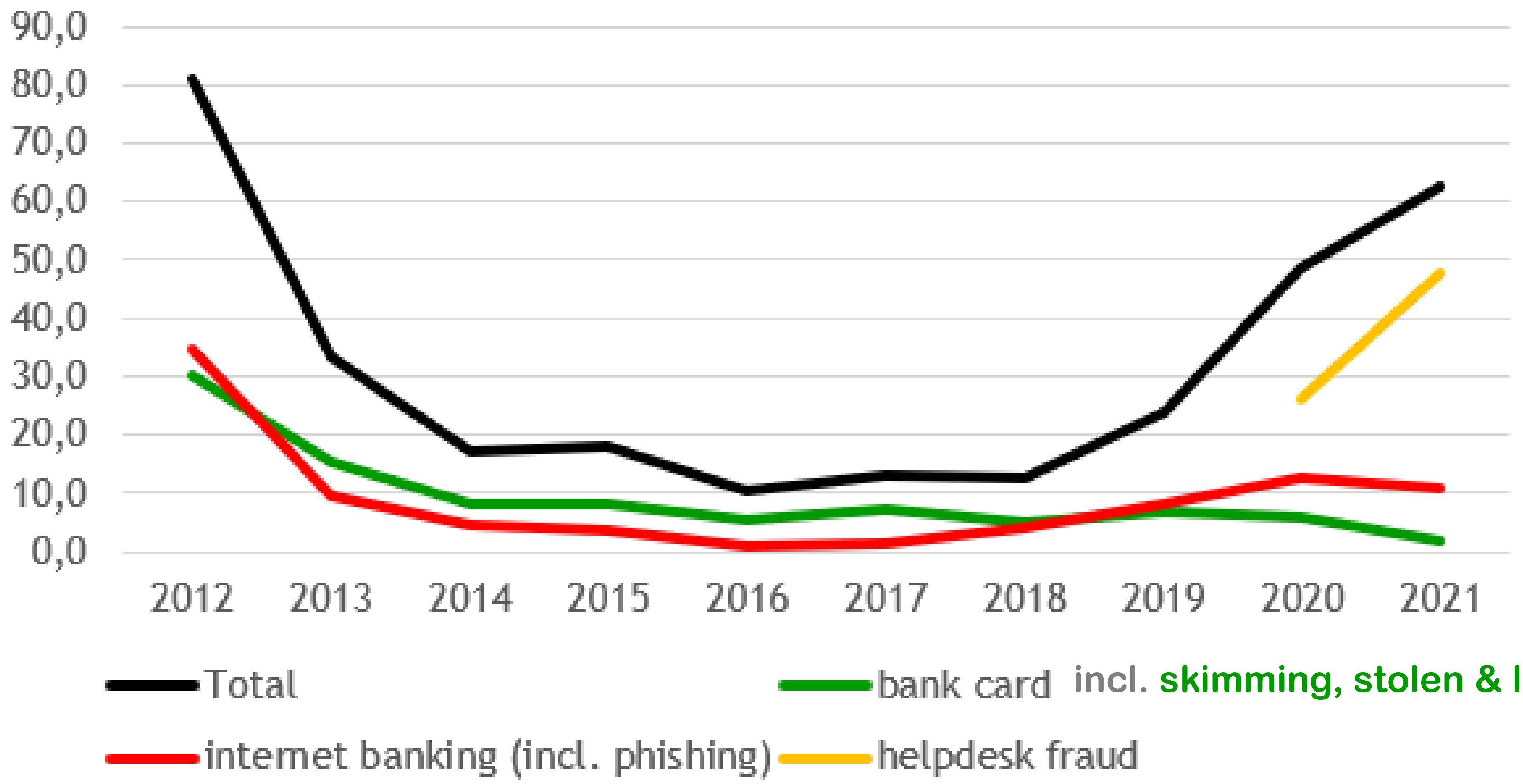
Radboud University Nijmegen

Payment fraud in Netherlands (Meuro)

Categorie	2018	2019	2020	2021
Internetbankieren: Phishing	3.936	7.938	12.837	10.847
Debitcard: Gestolen/verloren	3.322	6.607	5.846	1.911
Creditcard: Card not present	1.976	2.083	3.787	2.156
Bankhelpdeskoplichting (sinds 2020)	-	-	26.260	47.638
Totaal fraude[1]	**9.234**	**16.628**	**48.730**	**62.552**

Source: betaalvereniging.nl and nvb.nl

Payment fraud in Netherlands – longer term trends



incl. skimming, stolen & lost cards

incl. malware & phishing

Overview

- The EMV standard
  - Known issues with EMV
- EMV contactless
- Formalisation & Verification of EMV using F# and ProVerif
- EMV-CAP for internet banking
- Conclusions

EMV

- Started 1993 by EuroPay, MasterCard, Visa
- Common standard for communication between
  1. smartcard chip in bank card (aka ICC)
  2. terminal (POS or ATM)
  3. issuer back-end
- Specs controlled by EMVCo which is owned by
- Billions of cards in use
- Also contactless and on mobile phone

Motivation for EMV chip: skimming

Magnetic stripe (mag-stripe) on bank card can contain digitally signed information

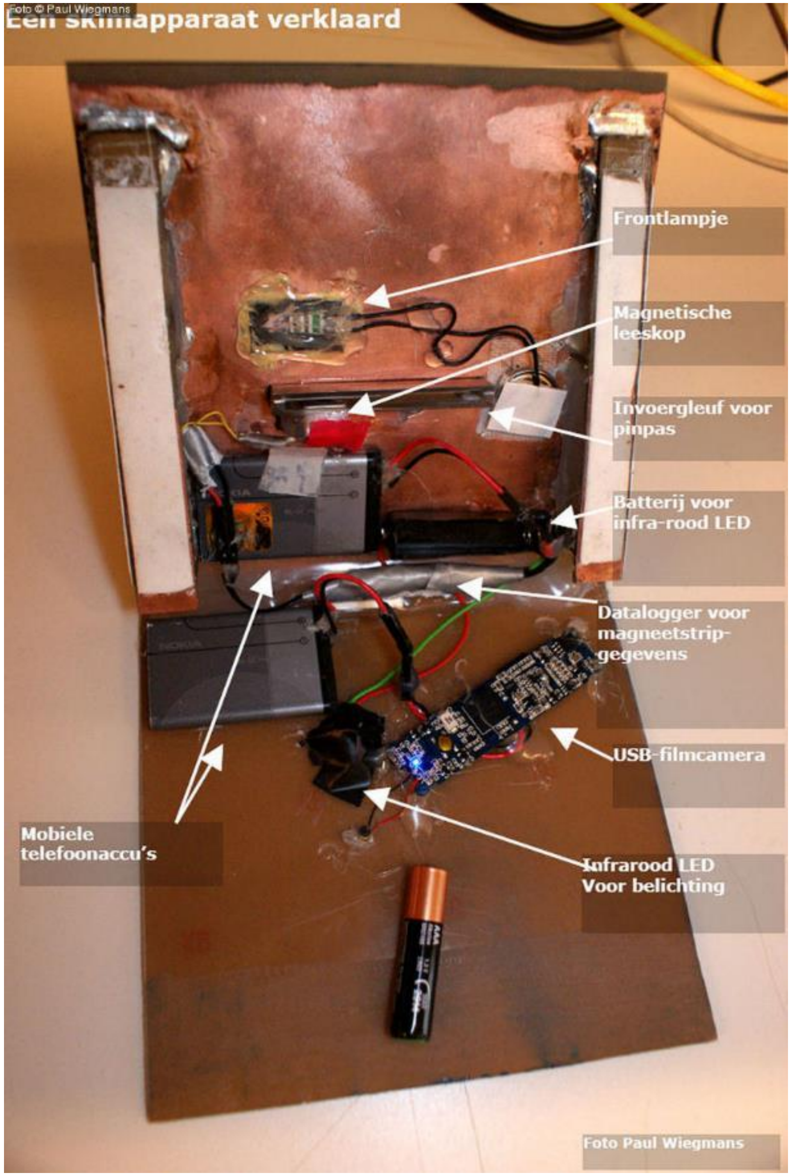
but… this info can be copied

Skimming equipment

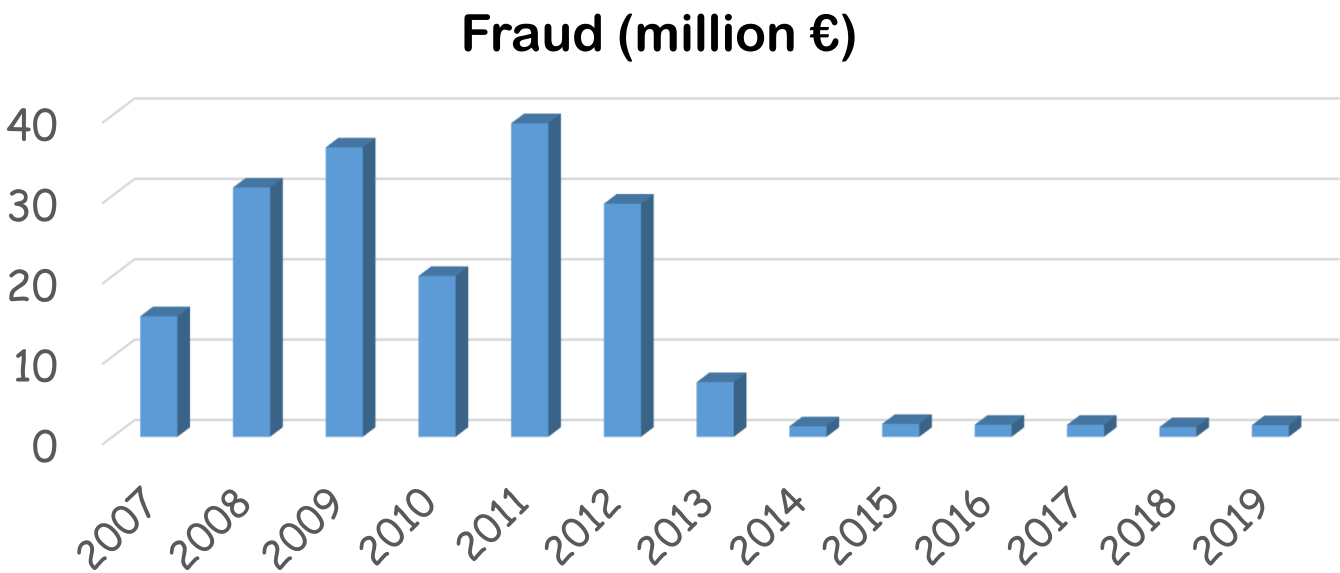
Fake keyboard
to intercept PIN code

Fake cover
that copies magnetic stripe

Skimming equipment for NS terminals



Skimming in the Netherlands



[Source: NVB/Betaalvereniging]

Drop due to

- **better monitoring, detection, and reaction (esp. blocking cards)**
- **introduction of EMV (2012)**
- **geoblocking (2013)**

Does EMV chip  reduce skimming?

- UK introduced EMV in 2006

	2005	2006	2007	2008
domestic	79	46	31	36
foreign	18	53	113	134

Skimming fraud with UK cards, in millions £ [Source: Payments UK]

- USA is still migrating to EMV, and criminals have moved there…

Liability shifts

Move to EMV chip involves liability shifts

- Customer liable for fraud with their PIN code
- Vendors liable for fraud if they still use magstripe

  In the USA, for POS starting Oct 2015, for ATMs Oct 2017, for petrol stations Oct 2020.

The EMV standard

The EMV protocol suite

- EMV is not a protocol, but a toolkit of building blocks for protocols with
  - 3 card authentication mechanisms
    - SDA, DDA, CDA
  - 5 cardholder verification mechanisms
    - online PIN, offline plaintext PIN, offline encrypted PIN, handwritten signature, no card holder verification
  - 2 types of transactions: offline, online

  All mechanisms again parameterised by Data Object Lists (DOLs)

- Specs public but very complex (4 books, >750 pages)
  - Specs do not motivate design or mention security objectives…

EMV protocol phases

I. Initialisation

Terminal reads some data from the card, incl. several DOLs

II. Card Authentication (using SDA, DDA or CDA)

III. Cardholder Verification (optional, for instance using PIN)

IV. Terminal & Card Risk Management

V. Transaction

where the card produces Application Cryptogram (AC)

with HMAC calculated with shared symmetric key 

*NB terminal does **not** have this key,*
so it cannot authenticate cryptograms when it is offline

Parameterisation using DOLs

- Data Object Lists specify a list of data elements
  - eg amount, currency, primary account number (PAN), application transaction counter (ATC), card/terminal-generated nonce (UN), ...

- Cards contain several DOLs that specify
  - data elements required as input to the card
  - data elements included in HMACs produced by the card

NB this means the protocol is still fully configurable. Eg including the amount and currency in the HMAC makes sense, but is not required.

EMV key set-up

1. Card & issuer have a shared symmetric key (3DES or AES) used to compute HMACs on transactions
   - Terminal does *not* have this key, so cannot check these
2. Issuer has private RSA key and terminal knows public key
   - This allows SDA: terminal authenticates *static* signed data on the card
3. (Optional) DDA & CDA cards have a private RSA key and associated certificate, signed by issuer
   - Card can now sign *dynamic* data that terminals can authenticate
     - to authenticate card *or* transaction

II. Card Authentication: SDA

1. SDA – Static Data Authentication

   - SDA card cannot do asymmetric crypto
   - Card presents static data (card no, expiry date etc) signed by issuer
     ie. card no, expiry date, ...++ { hash(card no, ...) }$_{\text{PRIVKEY-ISSUER}}$

   - Problem: can be replayed, so card can be cloned
     - Of course, clone will always say offline PIN check succeeded

   - Hence: *offline terminal can be fooled*
     - Transaction is signed (MACed) using symmetric key, but terminal cannot check this MAC
     - Issuer will spot this fraud later

- SDA is being phased out; Visa & Mastercard forbid issuance of offline capable SDA cards since 2011

II. Card Authentication: DDA

1. SDA – Static Data Authentication
2. DDA – Dynamic Data Authentication

- Card has (Pub,Priv) keypair and does challenge-response
  - This requires more expensive card than SDA: one that can do asymmetric crypto

- Security flaw : card authenticated, but *not* the transaction
- Hence: *offline terminal can still be fooled*
  - Attacker can let the terminal authenticate the card but then spoof the subsequent transaction data with its HMAC using some MitM device
  - Issuer will spot fraud later

II. Card Authentication: CDA

1. SDA – Static Data Authentication
2. DDA – Dynamic Data Authentication
3. CDA – Combined Data Authentication
    - Card has (Pub,Priv) keypair, as in DDA
    - Signature now added over all the transaction data
        - so now an offline terminal can check the authenticity of the card *and* of the transactions

II. Card Authentication

1. SDA – Static Data Authentication
2. DDA – Dynamic Data Authentication
3. CDA – Combined Data Authentication

- Most cards in use today are DDA

III. Cardholder Verification Methods (CVMs)

1. PIN

   a. online: PIN checked by the issuer

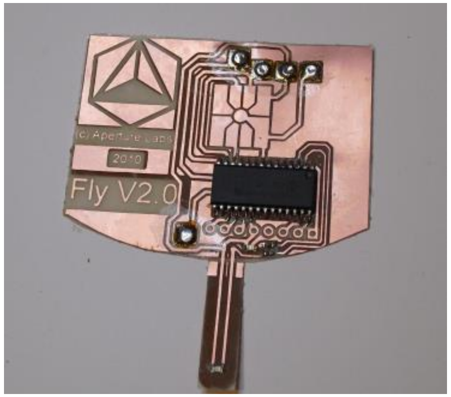
   b. offline: PIN checked by the chip

      b1. unencrypted

      PIN could be eavesdropped using shim

      b2. encrypted

      requires a card that can do asymmetric crypto



2. Handwritten signature

3. Nothing

NB: only offline PIN involves the smartcard chip; Dutch bank cards typically do online PIN

Cardholder Verification Methods (CVM)

- Terminal and smartcard negotiate which CVM is used
  - given their list of rules that specify allowed/supported method, in order of preference, with conditions

  Eg. transactions at tollroads do not require PIN,
  (contactless) payments under certain aim do not require PIN, ...

- Potential for trouble: forcing terminal/card to fall back to a weak CVM

conditions for applying specific CVM method

Value	Meaning
'00'	Always
'01'	If unattended cash
'02'	If not unattended cash and not manual cash and not purchase with cashback
'03'	If terminal supports the CVM [19]
'04'	If manual cash
'05'	If purchase with cashback
'06'	If transaction is in the application currency [20] and is under X value (see section 10.5 for a discussion of "X")
'07'	If transaction is in the application currency and is over X value
'08'	If transaction is in the application currency and is under Y value (see section 10.5 for a discussion of "Y")
'09'	If transaction is in the application currency and is over Y value
'0A' - '7F'	RFU
'80' - 'FF'	Reserved for use by individual payment systems

V. Transaction

For the transaction the card generates cryptograms

ie data with HMAC, and for CDA-cards, also a digital signature

- For offline transactions the card just generates one cryptogram (TC)

- For online transactions the card generates 2 cryptograms
  1. Card generates a first cryptogram (ARQC) that the terminal forwards to the issuing bank
  2. Bank sends a reply which the terminal forwards to the card
     - telling the card to go ahead or not
  3. Card generates second cryptogram (TC) confirming the transaction, provided the bank gave approval

V. Transaction

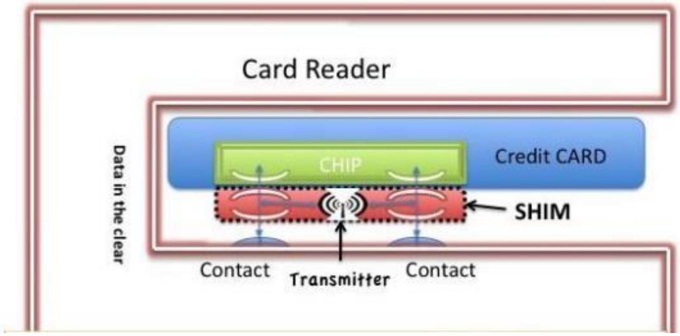
- The data is included in the cryptograms is configured by DOLs (Data Object Lists)
- It typically includes
  - the amount
  - a terminal-generated nonce (aka Unpredictable Number)
  - the card's Application Transaction Counter (ATC)
    - a counter that is increased with each transaction

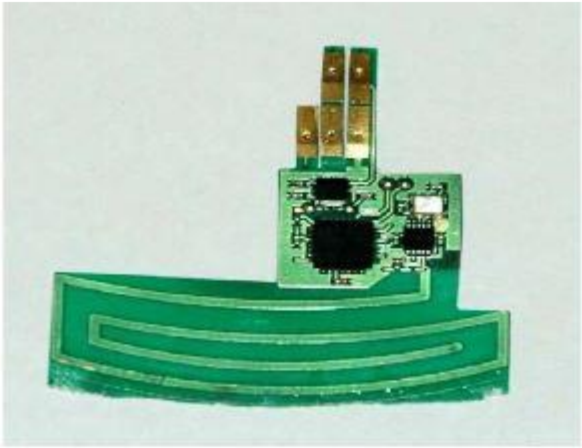
currency	amount	ATC	UN	HMAC = Enc(hash(currency, amount, ATC, UN))

EMV limitations & troubles…

Man-in-the-Middle attacks

Passive eavesdropping and active MitM possible with a shim





Two abuse scenarios

1. **tampering with a terminal**

   shim invisible in terminal for MitM attack

2. **tampering with a card, which is then used at normal terminal**

   eg acting as relay of (stolen?) genuine card to a terminal

Already discussed

1. SDA cards can be cloned
   - Fundamental limitation due to absence of asymmetric crypto on SDA cards
   - *NB back in the 1990s it was, but nowadays speed or costs are no longer valid excuses not to use ssymmetric*

2. DDA card cannot be cloned, but with a DDA card we can fool the terminal into accepting a bogus offline transaction
   - Stupid design decision to only use the asymmetric key to authenticate the card and not also the transaction

3. Backwards compatibility...

Track 2 magstripe data is also used by the EMV chip, so after eavesdropping on (unencrypted!) chip-terminal communication an attacker can reconstruct the magstripe

- If the card uses offline plaintext PIN, attacker can also eavesdrop the PIN, so attacker does not need a camera
- First incident with tampered EMV-CAP readers *inside Dutch ABN-AMRO bank branches*
  - Criminals caught & convicted in 2011
- EMV specs have been updated to avoid this

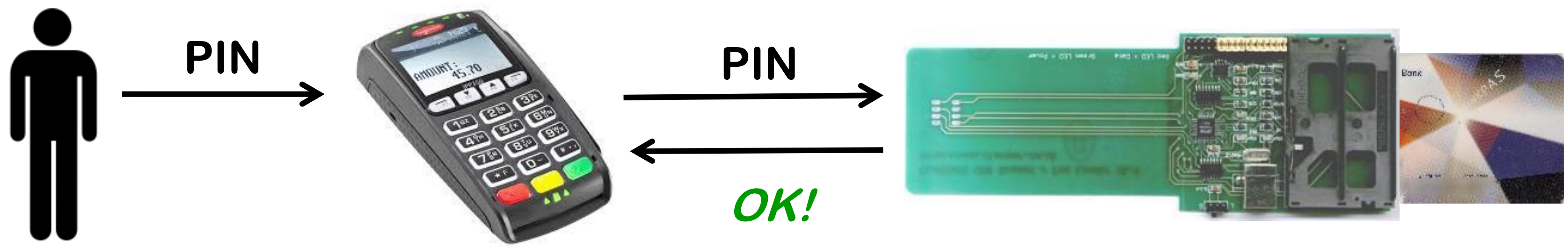
4. Offline PIN: *spot the security problem!*

Terminal can choose to do offline PIN, ie. ask the card to check the PIN

The OK response is simply the status word 0x9000

4. Offline PIN: *spot the security problem!*

Terminal can choose to do offline PIN, ie. ask the card to check the PIN



The OK response is simply the status word 0x9000

Problem: OK response is *not authenticated*
so terminal can be fooled by a Man-in-the-Middle attack

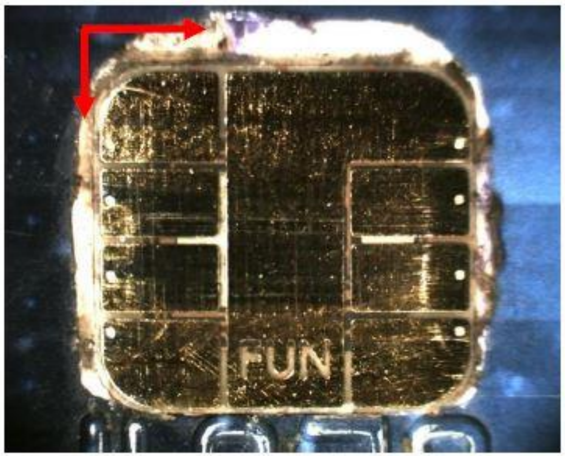
The cryptogram will reveal the transaction was PIN-less,
so the bank will later know the PIN was *not* entered

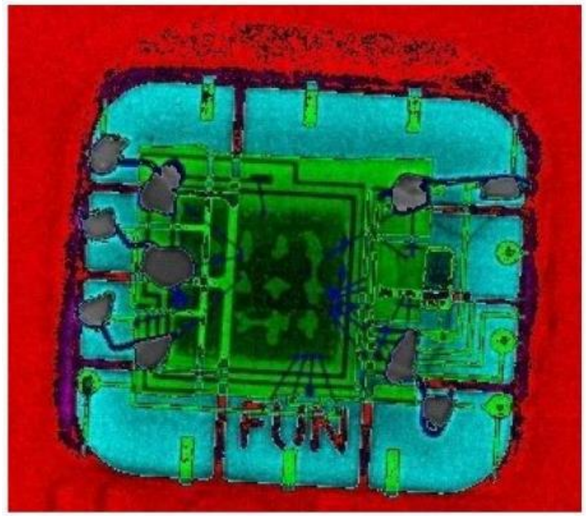
[Stephen Murdoch et al., *Chip & PIN is broken*, FC'2010]

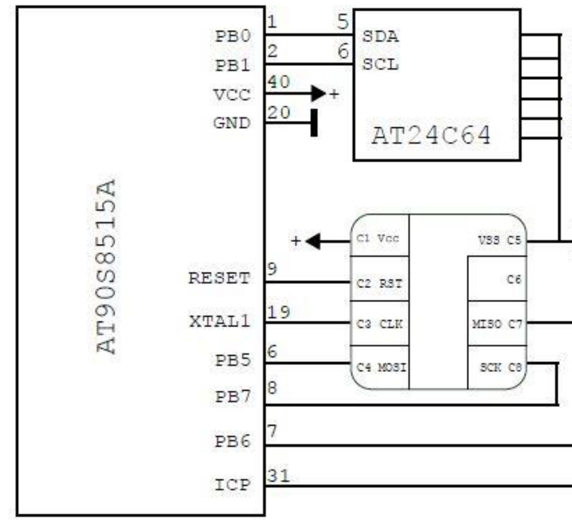
Reportedly won't work in NL, as Dutch cards always go online for PIN check

Criminal use of this ‘PIN OK’ attack

Tampered cards used by criminal gang: chips from stolen cards inserted under another chip that carries out MitM attack to fake ‘PIN OK’ response



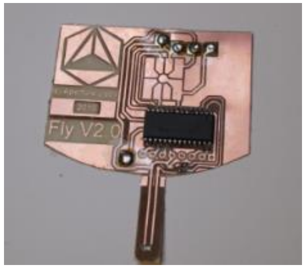




xray reveals
green stolen chip under
blue microcontroller

[Houda Ferradi et al., *When Organized Crime Applies Academic Results: A Forensic Analysis of an In-Card Listening Device*, Journal of Cryptographic Engineering, 2015]

5. Rollback to unencrypted PIN



- Shim can force a rollback to unencrypted PIN, by modifying card response to indicate the card does not support it
- Strangely, the terminal can tell the card is lying, as the signature over static card data is incorrect, but it does *not* abort the transaction!

[Barisani et at, *Chip & PIN is definitely broken,* DEFCON 2011]

- Impact limited because
  - just having the PIN of a DDA card is useless without the card
  - the attack is detectable in the back-end
- Reportedly, most terminals in NL patched to disallow this rollback
- We tried this attack, and bank detected it almost in real time
  - the one terminal we tried had not been patched…

6. Bad random number generation

Successive 32 bit random numbers in the log of a Maltese ATM

F1246E04

F1241354

F1244328

F1247348

This weak random number could be abused:

attacker with temporary access to card can copy static data and record enough responses to make a clone (pre-play attack)

[Bond et al. , *Chip and Skim: cloning EMV cards with the pre-play attack,* CHES 2012]

More information about criminal ATM hacks:

https://blog.kaspersky.com/sas-2017-atm-malware/14509, April 2017

https://darknetdiaries.com/episode/35/

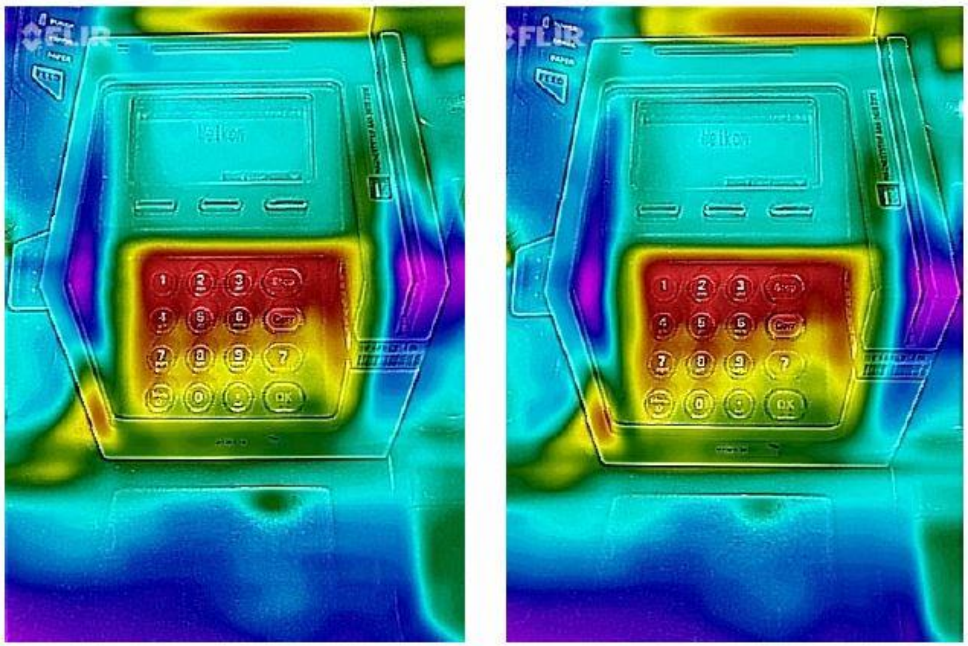
Stealing PIN codes using infrared?

Claims of attacks using infra-red camera to observe PIN

[Source: iPhone ATM PIN code hack, https://www.youtube.com/watch?v=8Vc-69M-UWk]

Stealing PIN codes using infrared?

These claims are bogus!



Thermal images we took after entering 2 different PINs

Dutch police and national TV programs (Tros Opgelicht & Opsporing Verzocht) believed this bogus story.

https://www.politie.nl/gezocht-en-vermist/gezochte-personen/2016/januari/09-oost-brabant/09-diefstal-pinpas-en-nieuwe-methode-pinpasfraude.html

For computer keyboards it has proven possible: 'Thermanator: Thermal Residue-Based Post Factum Attacks on Keyboard Data Entry', Asia CCS 2019, https://doi.org/10.1145/3321705.3329846

Inferring PIN code from (covered) hand movements

Machine Learning models can be trained to recover PIN code from movements of covered hand

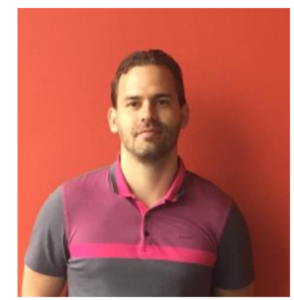
(a) *True digit = 7 Pred = 7 (0.999), 4 (0.000), 8 (0.000)*

(b) *True digit = 3 Pred = 3 (0.979), 2 (0.012), 6 (0.005)*

(c) *True digit = 6 Pred = 6 (0.819), 9 (0.170), 8 (0.009)*

(d) *True digit = 3 Pred = 3 (0.809), 2 (0.092), 5 (0.069)*

(e) *True digit = 3 Pred = 2 (0.329), 3 (0.315), 6 (0.185)*



Matteo Cardaioli, Stefano Cecconello, Mauro Conti, Simone Milani, Stjepan Picek, and Eugen Saraci
'Hand Me Your PIN! Inferring ATM PINs of Users Typing with a Covered Hand',
USENIX Security, 2022

Contactless payments

Contactless EMV

- with ISO/IEC 14443 contactless or dual contact card

  or NFC mobile phone

- Instead of one generic spec, as for contact payments, there are individual specs for each of the 10 versions
  - in 10 books, > 2000 pages

- Same building blocks as original contact spec, but some efforts to minimize the number of messages

Security challenge with mobile phones

1. *Where to securely store keys & PIN?*
2. *Where do compute MACs & signatures using these keys?*

Solutions include

1. **Use the SIM card**
   Briefly tried by Rabobank, but national scheme with all banks & telcos abandoned
2. **Using secure hardware in the phone: Apple Secure Enclave on iPhone, hardware-backed keystore (aka Strongbox Keymaster) on Android**
3. **Store keys in main memory & use the normal processor**

   Possible security enhancements:

   a) **using white-box crypto to obfuscate key material**

   b) **have symmetric key that can only be used for one transaction, so that app needs a new key for each transaction, aka EMV Tokenization**

Use of biometric authentication on phones can offer security advantage over smartcard.

Security & privacy worries

Contactless payments, without PIN, seem insecure…

- ***Who uses a metal container to shield their contactless bank card?***
- ***Who has asked their bank to disable contactless payments for their card?***
- ***Who thinks that contactless payments without PIN is less secure than contact payment with PIN?***

Passive attacks on contactless cards

- Eavesdrop on wireless communication between terminal & card
  - This is possible at 10-20 meters

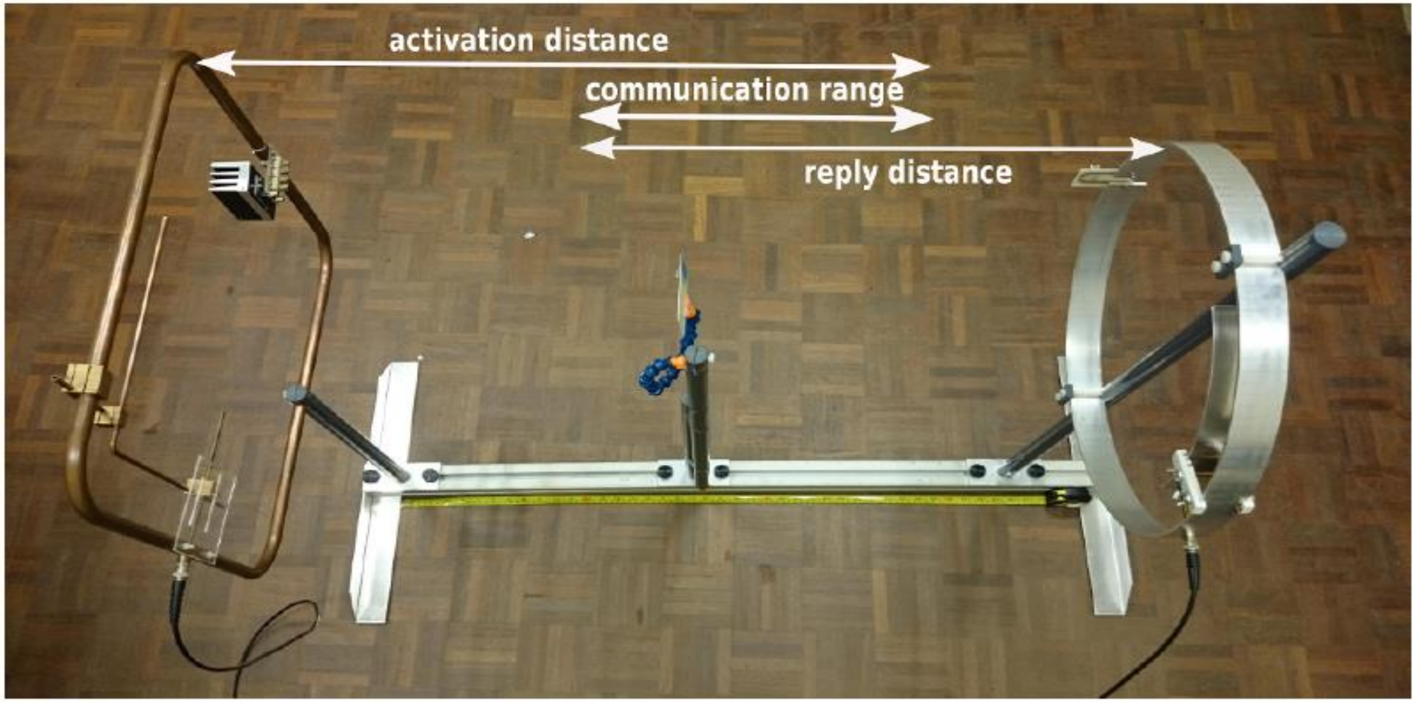
- Eavesdropping only poses a small privacy risk:

  The communication reveals eg. your bank account nr

  (Recall that most EMV communication is unencrypted)

Active attacks on contactless cards

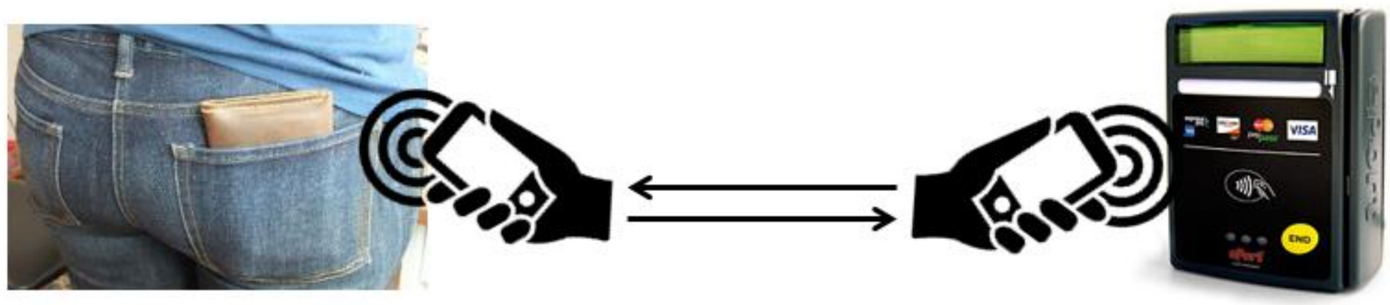
- Secretly activate card in someone's pocket (aka digital pickpocketing)
  - This is only possible at 40-50 cm
    because activating the card requires a strong magnetic field



[René Habraken et al.,
An RFID Skimming Gate Using Higher Harmonics, RFIDSec 2015]

Active relay attack on EMV contactless

- Active attacker can do a relay attack



- But is there a good criminal business model? Probably not…
- Relay attacks normally require very fast relay (< 200 msec) or else a time-out occurs.

  Time-out of contactless payment terminal:

  > 50 seconds

- Improvement in EMV protocol now includes distance bounding - ie. time-critical - step, but it will be many years before this ever gets implemented in cards & terminals

Risks of PIN-less contactless payments?

1. Risks of contactless payment *without* PIN
   a) You loose max. € 50 if your card is stolen
   b) You loose max. € 25 euro if you fall victim to a relay attack

   Dutch banks typically cover these losses.

2. Risks of contact payment *with* PIN
   a) You don't loose any money if your card is stolen
   b) You can loose €1000 or more if your card is stolen after attacker snooped your PIN code

   Banks will typically not cover these losses…

So the 'extra security' of the PIN probably *in*creases risk for customers.

As always: technical security weakness ≠ risk

where risk = likelihood x impact

Some flaws our students found

- Mistake in most first generation Dutch contactless cards:

  functionality to check the PIN code offline,
  which should only be accessible via the contact interface
  was also accessible via the contactless interface

  Possible risk for DoS attacks, rather than financial fraud?

  Flaw discovered by Anton Jongsma, Robert Kleinpenning, and Peter Maandag.

- Contactless payment terminals of one manufacturer could be crashed with a legal – but unusual – input
  - namely an extended length APDU

  Flaw discovered by Jordi van den Breekel

  Why are terminals not tested better as part of certification?

EMV contactless: BACKWARDS COMPATIBILITY

Early contactless cards suffered from two problems due to backwards compatibility problems

1. **Very early contactless credit cards reported magstripe data unencrypted over the air, so magstripe clone can be made**

   [Heydt-Benjamin et al, *Vulnerabilities in First-Generation RFID-enabled Credit Cards*, FC 2007]

2. **Later contactless credit cards use a dynamically generated 3 digit code to replace the 3 digit CVC code. But 3 digits is not a lot of entropy, so codes can be harvested & replayed**

   [M. Roland et al. *Cloning Credit Cards: A combined pre-play and downgrade*, WOOT 2013]

Formalising & Verifying EMV

[Joeri de Ruiter and Erik Poll, Formal analysis of the EMV protocol suite, TOSCA 2011]

Complexity of the EMV specs

Specs too complex to understand

- long specs, split over 4 books, > 750 pages
  - for contactless: another 10 books, > 2000 pages
- little or no discussion of security goals or design choices
- little abstraction or modularity

Problem: complexity

One sample sentence taken from these thousands of pages

> "If the card responds to GPO with SW1 SW2 = x9000 and AIP byte 2 bit 8 set to 0, and if the reader supports qVSDC and contactless VSDC, then if the Application Cryptogram (Tag '9F26') is present in the GPO response, then the reader shall process the transaction as qVSDC, and if Tag '9F26' is not present, then the reader shall process the transaction as VSDC."

Formalising EMV ?

- Can formal techniques for security protocol analysis with tools like ProVerif cope with EMV?

- First attempt: formalising EMV in ProVerif

  Horrible! Case distinctions in applied pi-calculus cause lots of duplication

  Beware: real protocols always involve multiple variants, so in ProVerif people typically only verify one variant, leaving out options & abstracting away from lots of messy details…

- Second attempt: formalising EMV in F#

  Much better! F# allows sequential if-statements & functions

Formalisation of EMV

(Known) security flaws can now be found automatically by FS2PV & Proverif tool for security protocol verification

Formalisation of EMV in F#

- EMV can be formalised in 370 lines of F# code
  - including all options
    - SDA, DDA, CDA
    - any card holder verification mechanism
    - off/online transations
  - But DOLs has to be fixed
    - Model uses minimal assumptions on DOLs taken from Dutch bank & credit cards
    - Hardcoded in the model, but could easily be changed

Part of EMV model: DDA

```
// Perform DDA Authentication if requested, otherwise do nothing
let card_dda (c, atc, (sIC,pIC), nonceC) dda_enabled =
  let data = Net.recv c in
  if Data.INTERNAL_AUTHENTICATE = APDU.get_command data then
    if dda_enabled then
      begin  let nonceT = APDU.parse_internal_authenticate data in
             let signature = rsa_sign sIC (nonceC, nonceT) in
             Net.send c (APDU.internal_authenticate_response nonceC signature);
             Net.recv c
      end
    else failwith "DDA not supported by card"
  else data
```

Properties checked with ProVerif

1. Sanity checks to ensure absence of deadlock
2. Secrecy of private keys
3. Highest supported card authentication method is used
    - eg no fallback to say SDA can be forced
4. ‘transaction security’: if a transaction is completed, then everyone agrees on the parameters (eg with/without pin, off/online, amount,…)

```
query   evinj:TerminalTransactionFinish(sda,dda,cda,pan,amount,…)
            ==>   evinj:CardTransactionInit(sda,dda,cda,pan,amount,…)
```

No new attacks found, but most existing attacks inevitably (re)discovered

EMV-CAP

EMV CAP protocol

- EMV chip used for internet banking or e-commerce
    - challenge-response mechanism using the bank card
- EMV CAP is defined on top of EMV:

an EMV-CAP session is an *aborted* EMV session, where one of the cryptograms is used to construct the 8 digit response

- internet banking
    - Mastercard : CAP (Card Authentication Program)
    - Visa : DPA (Dynamic Passcode Authentication)
- e-commerce
    - Mastercard: SecureCode
    - Visa: Verified by Visa
- *EMV CAP specs are secret but have been largely reverse-engineered*

Limitations of EMV-CAP

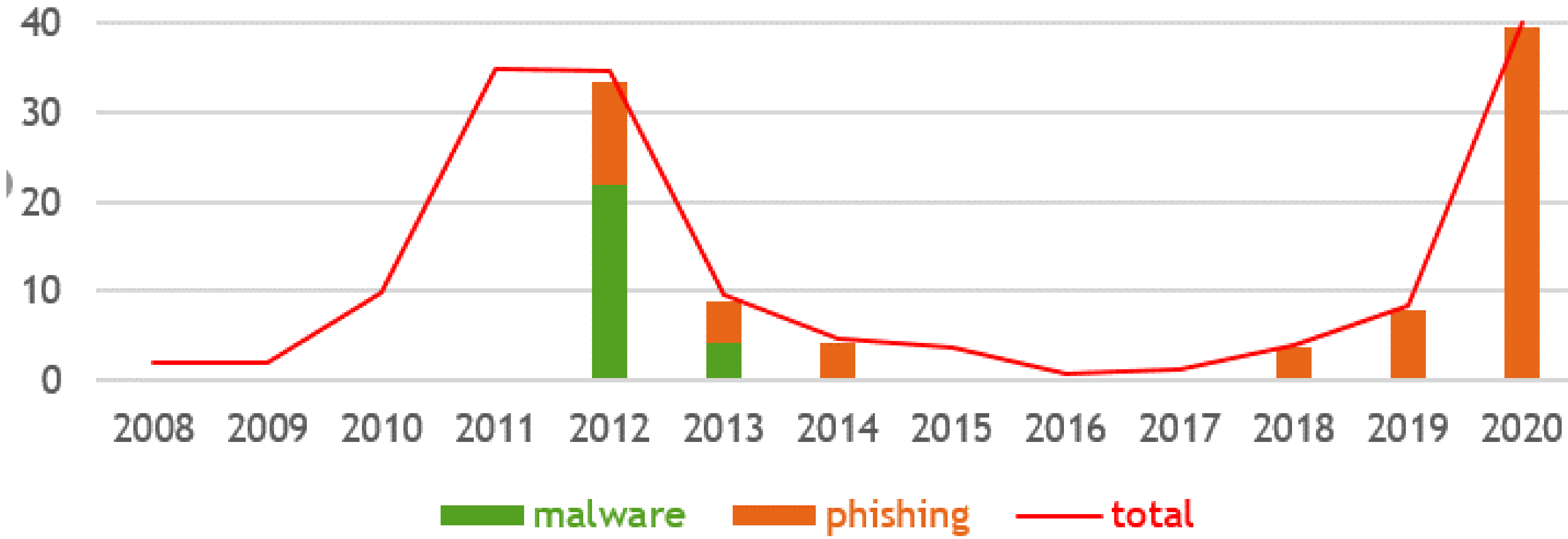
EMV-CAP does not protect against e.g.

- Man-in-the-Browser attacks,

  ie. malware inside the browser or on the user's PC

- Phishing attacks tricking customers to go to fake bank websites
- Social engineering attacks by telephone on customers

Internet banking fraud in Netherlands (millions euro)



After 2012, up to 2019, fraud under control thanks to

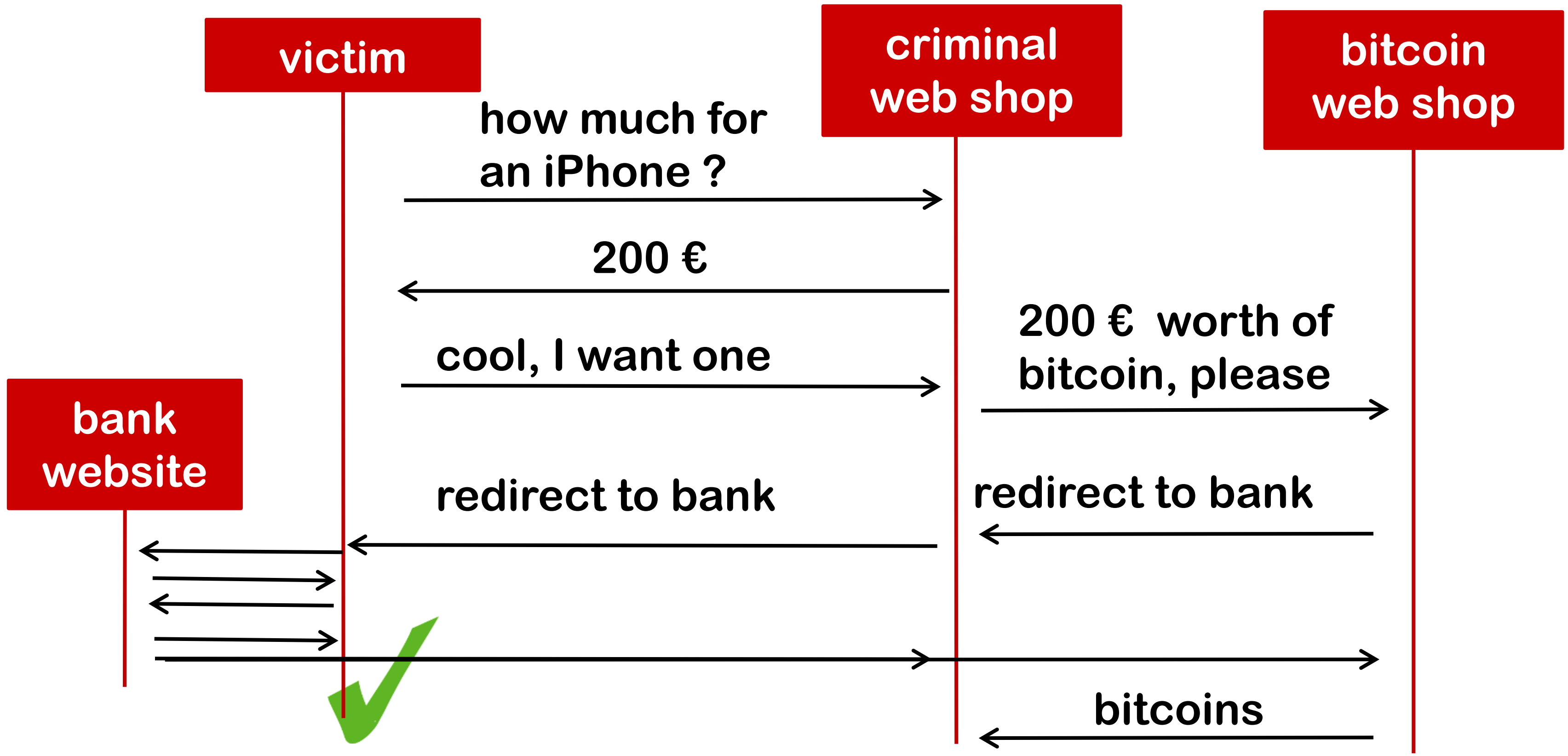
1. **better monitoring - for suspicious transactions & money mules**
   - **finding money mules, to extract money from the system without being caught, is the bottleneck for attackers**
2. **awareness campaigns**
3. **criminal switching to ransomware as better business model?**

[Source: Betaalvereniging]

Example attack on internet banking (1)

- Your online bank statement shows you received 3000 euro from some company you never heard of
- You get a phone call from the bank, saying that this is a mistake and asking you to transfer the money back
- You never received 3000 euro, but malware in your browser inserts the fake transaction
  - i.e. Man-in-the-Browser attack
- When you transfer the money back, that is not a fake transaction…

Example attack on internet banking (2)



- Problem: Money trail no longer leads to criminal webshop, but to the innocent bitcoin shop
- Root cause: messages to user not very informative, so user does not spot the attack
- Solution: better monitoring, and banks impose extra rules on bitcoin shops & online casinos for allowing internet payments

Protocol flaw in EMV-CAP Mode 2

1. user → reader : challenge
2. reader → card : 0x000000
3. card → reader : K ,

   where $K = HMAC_{Key}(0x000000 ++ counter)$
4. reader displays some digits from {challenge}_K

So challenge C never goes to the card!

The message in step 2 is predictable so an attacker with temporary access to a card could harvest responses K to do internet banking later

[P. Szikora and P. Teuwen, Banques en ligne: à la découverte d'EMV-CAP, MISC (Multi-System & Internet Security Cookbook) , 2011]

Example attack on internet banking (3)

- Security flaw in Gemalto e.dentifier2 for ABN/AMRO
  - only when device is used with USB cable
- Found during Master thesis project of Arjan Blom

  [A. Blom et al.,
  Designed to Fail: A USB-Connected Reader for Online Banking
  NordSec 2012]

- Bug now fixed, but old vulnerable devices not recalled

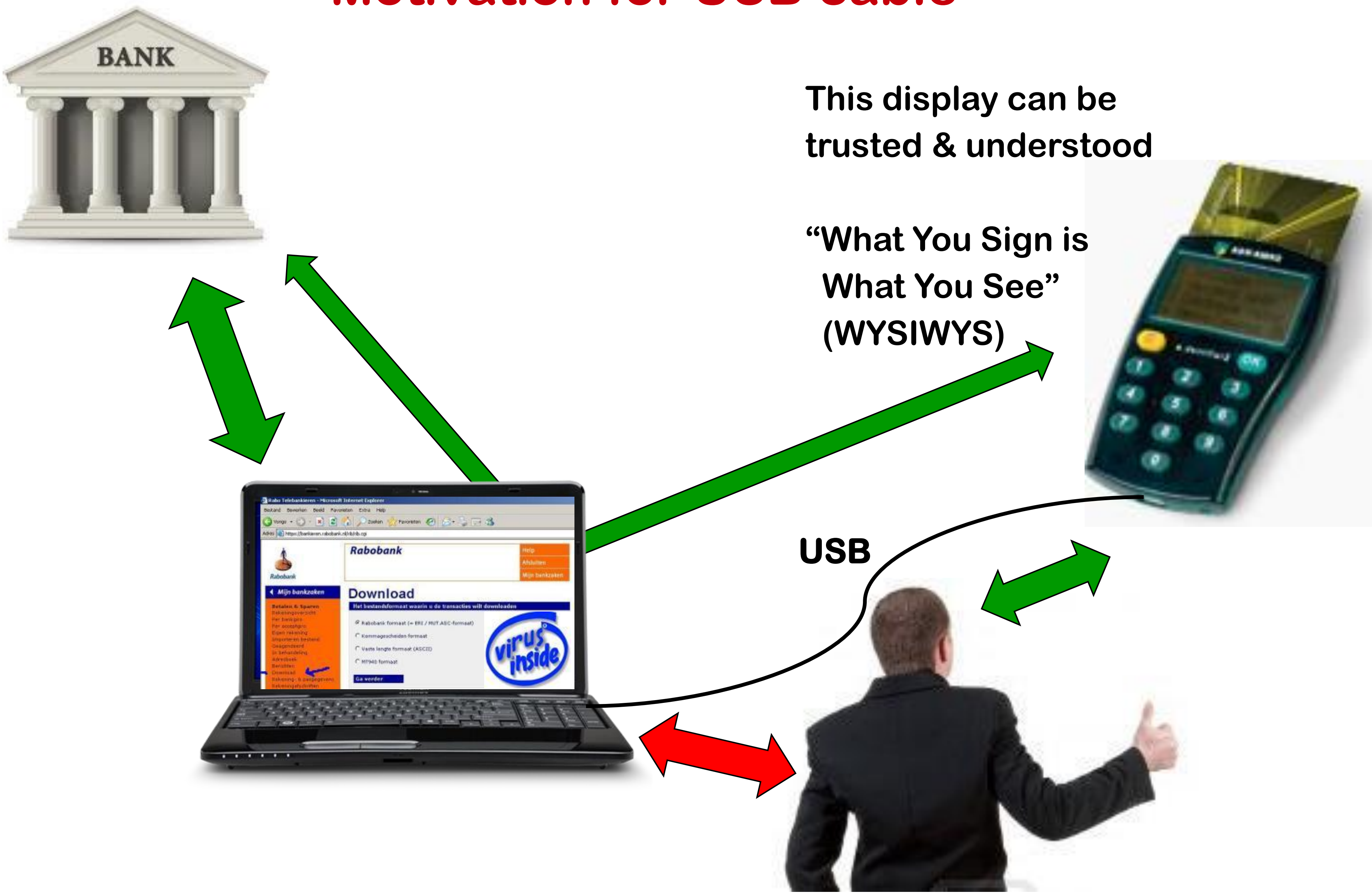
Motivation for USB cable

This reader can be trusted.
But can the user understand the semantics of numbers?

Computer display of cannot be trusted (despite)

→ 23459876
← 123654

Motivation for USB cable



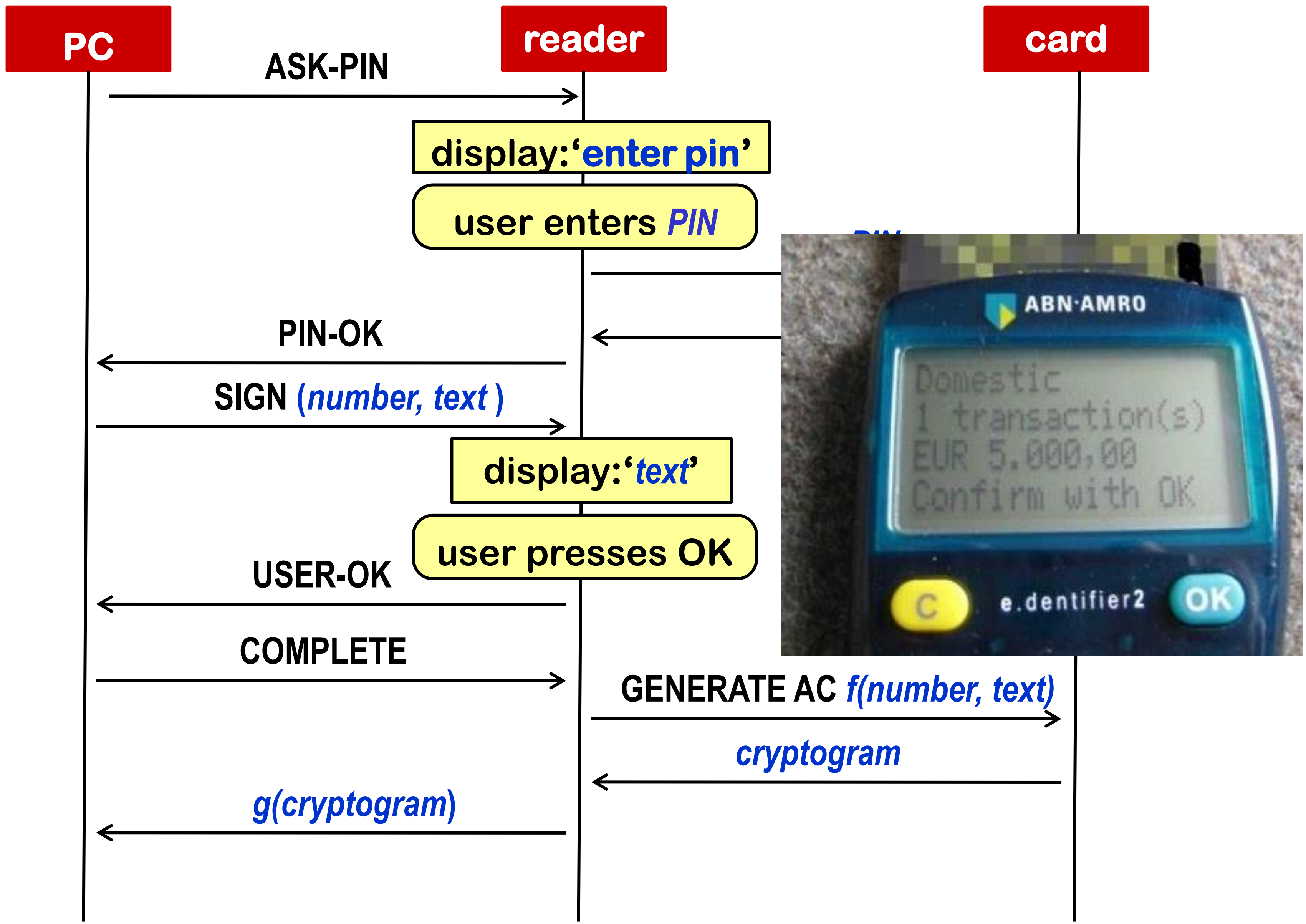
Rabo Scanner

- Alternative solution to allow communication to hand-held reader (with coloured QR code)
- No communication back to the PC, unlike with USB cable

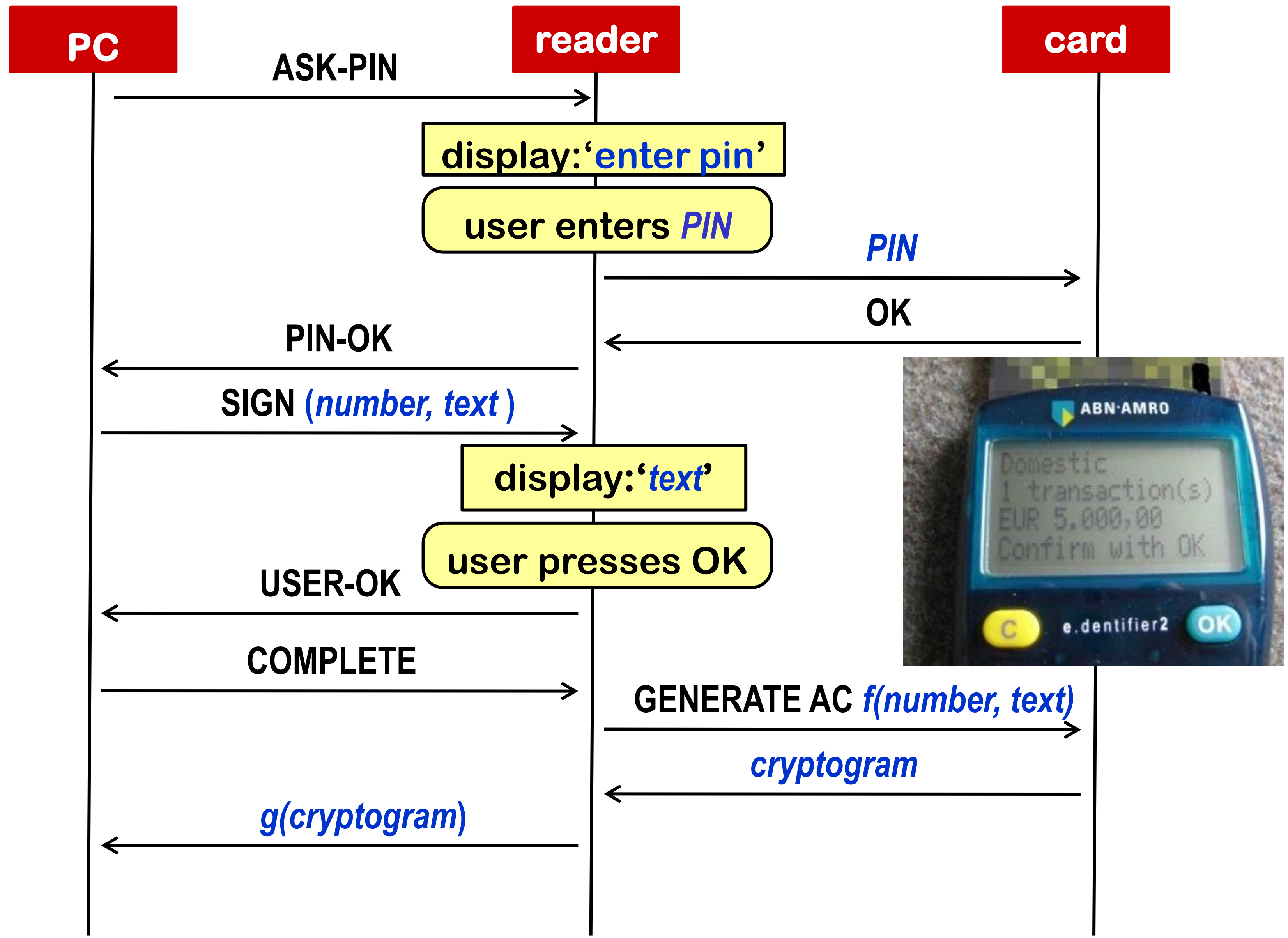
Analysis of e.dentifier: first observation

- **Text for display goes in plain-text over USB line**
- **So malware on the laptop can make the token show any message**

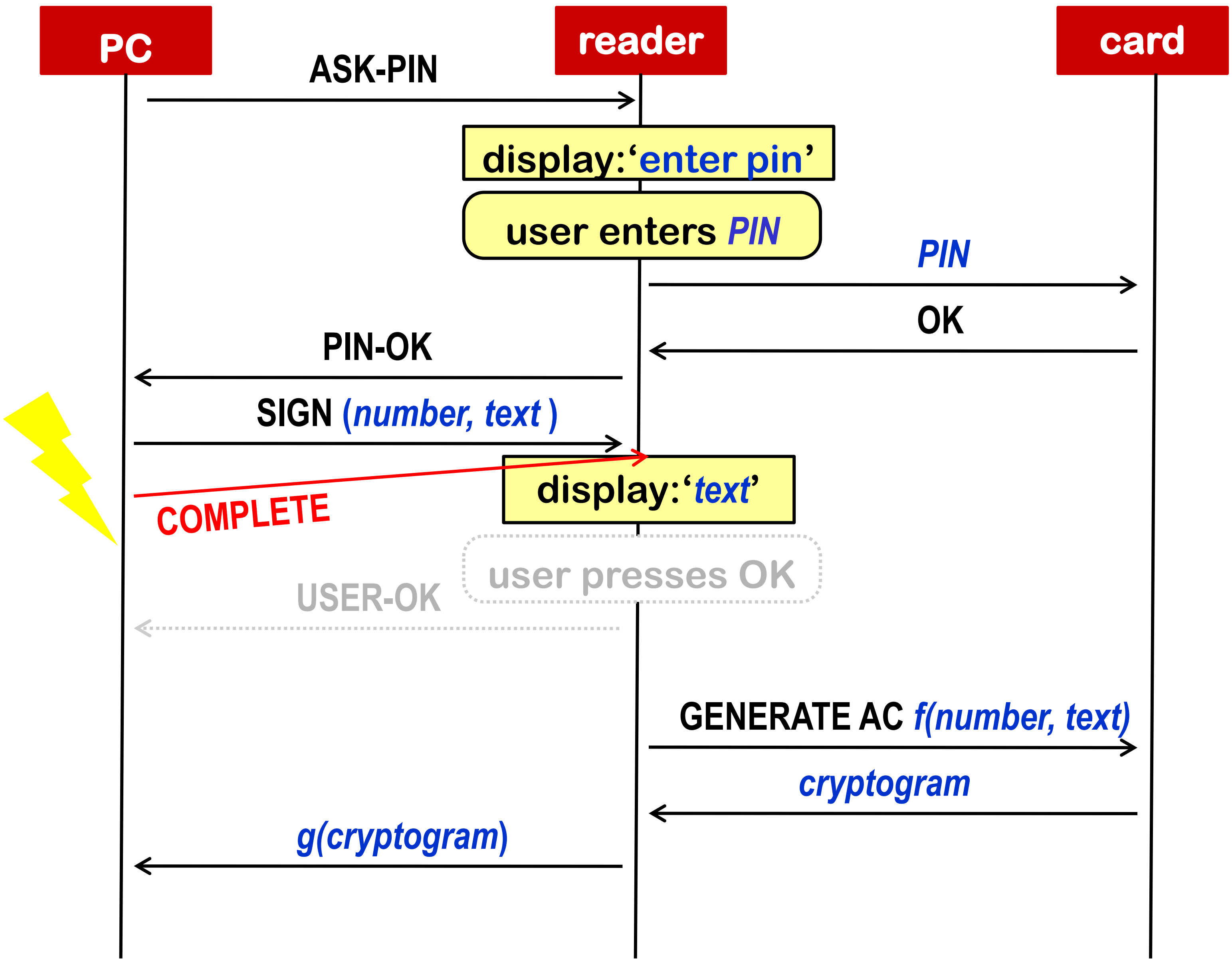
Reverse-Engineered Protocol



Reverse-Engineered Protocol



Attack!


PC
reader
card
ASK-PIN
display:'enter pin'
user enters PIN
PIN
OK
PIN-OK
SIGN (number, text)
COMPLETE
display:'text'
user presses OK
USER-OK
GENERATE AC f(number, text)
cryptogram
g(cryptogram)

Problem with Todos/Gemalto e.dentifier2

It's possible to press OK via
USB cable...

Malware on an infected PC could change all the transaction details and press OK

Purely academic, no criminal ever abuses thus

[Arjan Blom et al.,
Designed to Fail: A USB-Connected Reader for Online Banking, NordSec 2012]

Conclusions

Conclusions about EMV & banking world

- EMV protocol suite is way too complicated
  too many options, written down in confusing way, without useful abstractions, without explaining security, …
- Banks - or their suppliers - routinely screw up security. Eg we saw
  - DDA: why not let the card sign transactions if it can do RSA?
  - backwards compatibility problems
  - lousy random number generators in ATMs
  - misconfiguration of contactless cards
  - contactless terminal crashing on extended length APDUs
  - protocol flaws in EMV-CAP mode 2 and e.dentifier2
  - …
- Technical flaws harmless if there is no good attacker business model. But always a public relations risk.
- Bottleneck in security here: AUTHENTICATION

Conclusions about the banking world

- Not so clear who is taking responsibility for checking security
  - The banks?
  - Scheme holders such as MasterCard and Visa? EMVco?
  - Their suppliers? (eg Gemalto, ST Microelectronics,...)
  - The parties doing certification tests for scheme holders? (eg UL)
  - The Dutch or European Central Bank?
- Banks appear to assume - and trust - that others check the security!
- Or maybe their employees are happy with Cover-Your-Ass security?

Moral of the story

- Keep it simple!
- Protocols should only have *one* version/variant, namely the secure one!
- Never assume that somebody else (eg. a vendor, Mastercard, Visa, ...) has checked that things are secure!

Possible research ideas

- **What would a post-quantum version of EMV look like?**

  The old-fashioned reliance on a shared symmetric key (still 3DES in some bank cards!) may turn out to be an advantage…

- **How do the security levels of mobile phone-based alternatives compare to smartcards?**