Reverse engineering Internet banking

Eduardo Pablo Novella Lorente

e.novellalorente@student.ru.nl
http://ednolo.alumnos.upv.es

Institute for Computing and Information Sciences – Digital Security
Radboud University Nijmegen

24 June 2013
Nijmegen (The Netherlands)
Introduction

1. Handheld smartcard readers: USB-connected & unconnected
2. ABN-AMRO & ING Direct
3. E.dentifier2: Attack to e.dentifier2 (2012)
4. Try the attack in the new readers
5. Additional functionalities: Mode1 & Mode2
Background

**EMV-CAP**
- Based on EMV
- Reverse engineered
- EMV-CAP handheld smartcard readers
- Login & Signing using challenge-response

**e.dentifier2**
- ABN-AMRO EMV-CAP reader
- Reverse engineered by Digital Security
- Modes: USB-connected & unconnected
- Operations: Login & Signing of transactions
Background

Challenge-response

2 Application Cryptograms (AC) are created as proof of authorization from smartcard

- ARQC (Authorization Request Cryptogram). Response against the challenge sent
- AAC (Application Authentication Cryptogram). Verification

DigiPass 850

- ING Direct EMV-CAP reader
- Modes: USB-connected & unconnected
- Operations: Login & Signing of transactions
Background

SWYS

- aka "What You Sign Is What You See" (WYSIWYS)
- Pretend to avoid Man-in-the-browser attacks
- PIN code has been entered in the reader
- Cardholder can accept/deny operations' messages
- Cardholder can understand messages
- But: Bad designed (Attack by Digital Security)
Vulnerability in the old e.dentifier2

Diagram:
- PC
- Old e.dentifier2
- Smartcard

1: ASK PIN
2: VERIFY pin guess
3: PIN OK
4: PIN OK
5: SIGNDATA number
6: SIGNDATA text
7: user pressed OK
8: GENERATE_AC
9: GENERATE_AC f(text,number)
10: ARQC
11: GENERATE_AC f(text,number)
12: AAC
13: g(ARQC)
Attack in the old e.dentifier2

1: ASK PIN
2: VERIFY pin guess
3: PIN OK
4: PIN OK
5: SIGNDATA number
6: SIGNDATA text
7: GENERATE_AC
8: GENERATE_AC f(text_number)
9: ARQC
10: GENERATE_AC f(text_number)
11: AAC
12: g(ARQC)
Patch in the new e.dentifier2

1: ASK PIN

ENTER PIN: 

USER TYPES PIN

4: PIN OK

5: SIGNDATA number

6: SIGNDATA text

DISPLAY TEXT

7: GENERATE AC

8: ABORT PRESS ‘C’

2: VERIFY pin guess

3: PIN OK
Possible correct SWYS protocol

1: ASK PIN
2: VERIFY pin guess
3: PIN OK
4: PIN OK
5: SIGNDATA number
6: SIGNDATA text

PC

Possible identifier2

ENTER PIN:

USER TYPES PIN

smartcard

DISPLAY TEXT

USER PRESSES OK

7: GENERATE_AC f(text,number)
8: ARQC
9: GENERATE_AC f(text,number)
10: AAC
11: g(ARQC)
Tools

1. Wireshark & USBTrace
2. RebelSim & RealTerm
3. Fake bankcard with Javacards
4. Own webpage
5. Python code using PyUSB library
6. Firebug Add-on
Big picture

- **HTTP(S)**
  - GET
  - POST
  - Eavesdropped using: Proxy
  - Firebug

- **USB**
  - bulk data transfers
  - Eavesdropped using: Wireshark
  - USBTrace

- **ISO7816**
  - APDU
  - AID
  - Eavesdropped using: RebelSim

---

**Eduardo Pablo Novella Lorente**

Bachelor thesis

Reverse engineering Internet banking

Radboud University Nijmegen
Mode1

GetMode1Response (Challenge, Currency, Amount)

JavaScript functions in ABN-AMRO website. File: \textit{BECON.js}

- Reverse engineered
- Signing using challenge-response
- Unconnected mode has this mode
- Challenge 8 numeric digits
- Currency 4 digits for EMV code (0978 €) (0826 £) (0840 $)
- Amount 12 numeric digits between [0000.000.000,00 - 0999.999.999,99]
Protocol of GetMode1Response

1: \text{GetMode1Response('x0\ldots x0', 'y0\ldots y0', 'z0\ldots z0')}\n
2: USB command
   Payload 32 bytes
   00 06 01 15 9F 37 04 00 06 x3 x3 x3 x3 09 02 06 00 06 z3 z3 z3 z3 z3 z3 z3 z3 00 05 5F 2A 02 y3 y3 y3 z3 z3

3: VERIFY pin guess

4: PIN OK

5: \text{GENERATE\_AC f(x,y,z)}

6: ARQC

7: \text{GENERATE\_AC f(x)}

8: AAC

9: USB response: User press OK
   Payload: Response
   00 04 R\ldots R\ldots R\ldots 00 00

10: Status=0
    Response=R\ldots R\ldots R\ldots R\ldots R\ldots R\ldots R\ldots R\ldots R\ldots
Reverse engineering Mode1

Confirm currency and amount:
€ 999,999,999.99
GetMode2Response()

JavaScript functions in ABN-AMRO website. File: BECON.js

- Reverse engineered
- Login
- Generate a right response
Protocol of GetMode2Response

1. Web Browser
   - GetMode2Response()

2. PC
   - USB command
     - Fixed payload 8 bytes
     - 00 01 02 00 00 00 00 00

3. e.developer
   - ENTER PIN: _

4. smartcard
   - USER TYPES PIN
   - VERIFY pin guess
   - PIN OK
   - GENERATE AC
   - ARQC
   - GENERATE AC
   - AAC

9. USB response: User press OK
   - Payload: Response
     - 00 04 RR RR RR RR RR 00 00

10. Status=0
    - Response=RR RR RR RR RR RR
• if (SWYS) safe++; else problems=true;
• Mode1 & Mode2 are more secure