

# Smartcards

## ISO 7816 & smartcard operating systems

Erik Poll

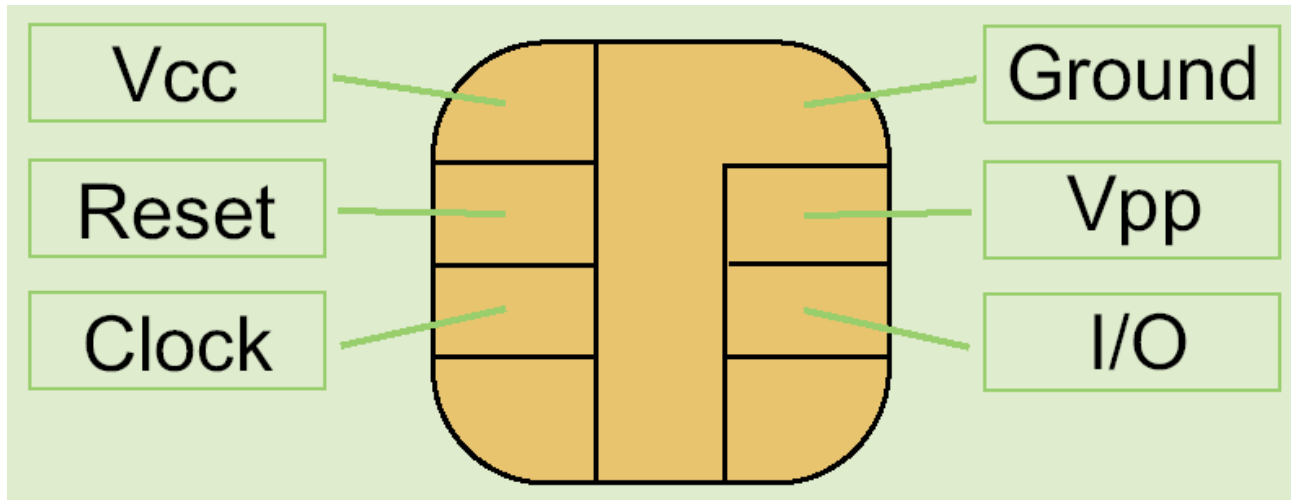
Digital Security

Radboud University Nijmegen

# Standard for contact smartcards ISO 7816

- 7816-1 Physical characteristics
- 7816-2 Dimension & size of contacts
- 7816-3 Electronic signals and transmission protocols
  - defines voltage & current requirements
- 7816-4 Inter-Industry Commands for interchange
  - standard set of commands
- 7816-5 Numbering system and Registration Procedure for Application Identifiers (AIDs)
- ....

## Contact cards (ISO 7816-2)



### External power supply and external clock

- Originally 5 V, now also 3V or 1.8V
  - 3.5, 5, 10 MHz
- Vpp - higher voltage for writing EEPROM - no longer used as it introduces a serious security weakness

# Smart card terminals

- Terminal (aka CAD, Card Acceptance Device) and smartcard operate as Master & Slave
  - smartcard cannot initiate actions

# The Terminal Problem!

- no trusted I/O between user and card
  - no trusted display
  - no trusted keyboard
  - why is this a problem?
  - some experimental cards with displays, keyboards, or fingerprint readers.

## Card Activation (ISO 7816-3)

### 1. terminal activates card

- earth; voltage; clock; reset

### 1. card responds with ATR (Answer To Reset)

- max 33 bytes, usually a lot less (for speed)
  - obligatory info about the protocol used, eg
    - T=0 byte-oriented
    - T=1 block-oriented
  - usually some manufacturer info
    - id of OS and version no. of ROM mask
  - obligatory last byte XOR checksum
  - supported baud rate for I/O
- must be sent between 400 & 40,000 clock cycles

## APDU communication (ISO 7816-4)

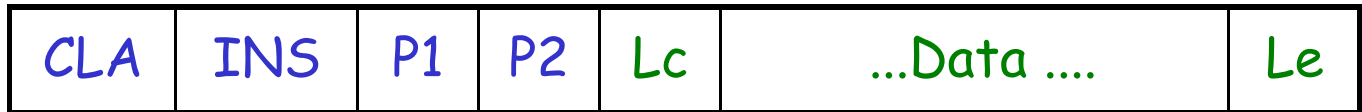
All subsequent communication via APDUs

Application Protocol Data Units

which are just sequences of bytes in particular format

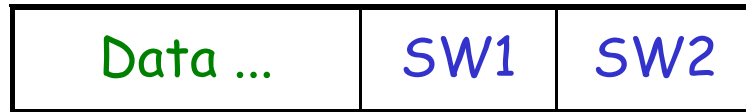
1. Terminal sends command APDU
  2. Card replies with response APDU
- etc, etc ....

# Command APDU



- CLA class byte
  - INS instruction byte
  - P1,P2 parameters
  - Lc length of data block
  - Data Lc bytes of data
  - Le length of expected response
- } obligatory
- } optional

## Response APDU



- Data : Le bytes of data (optional)
- SW1, SW2 : status word (obligatory)

# APDU coding conventions

- Some conventions for CLA, INS etc. are given in ISO 7816-4
- Conventions for status word SW1 SW2
  - normal processing 61xx, 9000
  - warning processing 62xx, 63xx
  - execution error 64xx, 65xx
  - coding error 67xx, 6Fxx

# Logical channels

- Modern cards provide several logical channels to talk to multiple applications on the card concurrently
  - eg mobile phone talking to phone book *and* organiser stored on SIM card

# Future

- ISO7816 protocol stems from 80s and it shows slow speed & small size of APDUs a bottleneck!
- faster communication speeds wanted
  - eg. USB 2.0
- more modern protocols wanted
  - http(s)

# Smartcard software and Operating Systems

# Smartcard operating systems

- Similar evolution as for normal OSs, but faster
  - additional factor: storing code in ROM vs EEPROM
- Still *very* primitive compared to normal OSs such as Windows or Linux/UNIX
  - no multi-programming, hardly any I/O, ...
    - but multi-threading in newest JavaCard 3.0...

# Smartcard OS

## Tasks:

- life-cycle management
  - of card + individual applications
- instruction processing
- memory management
- I/O
- hardware error handling
  - incl. support for atomic EEPROM updates
    - needed because the possibility of power failure by card tear

# Smartcard life cycle (ISO 10202-1 - cancelled)

- Production of chip & card
  - testing & removing test functionality
- Card preparation
  - completing OS
- Application preparation incl. Personalisation
  - initialising applications
  - personalisation / individualisation
    - both electrically & optically
- Card utilisation
  - (de)activation of applications
- End of card utilisation
  - de-activating applications
  - de-activating cards

# Smartcard OS evolution

1. no OS: one application, burnt into ROM
2. standard libraries in ROM, applications in EEPROM
3. proprietary operating systems
  - programs written in machine code or C
  - providing standardised file system (ISO7816-4) with access control
1. modern multi-application smartcards
  - MULTOS
  - JavaCard
1. next generation, experimental 'concept' card
  - JavaCard 3.0

## OS "completion"

1. Initially, card contains ROM mask
2. Simple loader in ROM executed to load EEPROM
3. Checksum computed
4. Switch to mode where code in ROM and EEPROM can be executed

# ISO 7816-4 file system

- MF (Master File) - ie. root directory
- DF (Directory Files)
- EF (Elementary Files) - external or internal
  - internal = for use of OS only
  - external = for use of applications
- file formats:
  - binary
  - linear fixed sized records
  - linear variable sized records
  - cyclic fixed sized records (why?...)
    - useful for logging
    - useful to wear out EEPROM evenly

# File attributes

Also includes more baroque things such as

- **WORM (Write Once, Read Many time)**
  - realised in hardware or software
- **EDC (Error Detection Code)**
- **atomic write access**
- **multiple storage attribute**
  - for frequently used files, to prolong lifetime of file given limited EEPROM life
- **data transfer selection attribute**
  - on dual-contact cards, to make file accessible only via contact or contactless interface

# ISO 7816 commands

- ISO 7816 defines a set of **standard commands** for
  - **file system access and management**
  - **PIN codes**
  - **authentication by challenge-response**
  - **crypto**
- Related standards, that build on top of this
  - **EMV**
  - **GSM 11.11** and its superset **EN 726-3**
  - **ICAO e-passport**

# Typical application life cycle

- installation of application (aka applet)
  - uploading & installing code
- personalisation
  - uploading application data
  - afterwards, application starts in normal active life
- end-of-life
  - disabling all functionality
  - possibly leaving logging functionality enabled
  - upon external command or because the card notices something fishy going on

# File access commands (ISO 7816)

Standard commands for

- reading & writing
  - eg **READ BINARY, READ RECORD, ...**
  - increase & decrease by n
    - for cyclic files
- append
- delete
- lock & rehabilitate
- identification / authentication

# Identification command (ISO 7816)

- VERIFY command
  - for PIN code verification
    - CHV = Card Holder Verification = PIN
  - also used for verification of biometric

# Authentication commands (ISO 7816)

- using challenge-response
- authentication of *card*
  - INTERNAL AUTHENTICATE
    - arguments: random, algorithm , key no
    - card returns:  $\text{enc}(\text{key}, \text{random})$
- authentication of *terminal*
  - GET CHALLENGE
    - card returns random number
  - EXTERNAL AUTHENTICATE
    - arguments:  $\text{enc}(\text{key}, \text{random})$ , algorithm, key no)

# Authentication (ISO 7816)

- *mutual* authentication
  - GET CHIP NUMBER
    - card returns chip number
  - GET CHALLENGE
    - card returns smart card random s\_rnd
  - MUTUAL AUTHENTICATE
    - arguments: enc(key, terminal random, s\_rnd, chip number), algorithm, key no
    - card returns: enc(key, terminal random, s\_rnd)

# Modern multi-application cards

- Downloadable program code, in high level language
  - multi-application
  - post-issuance download
- Examples
  - MULTOS
    - first of these "modern" smartcard OSs
  - JavaCard
    - esp. popular as GSM sims
  - Windows for Smartcards †
    - since abandoned

# Modern multi-application cards

## pros

- **vendor-independence**
  - old cards have proprietary OSs and instruction sets
- **fast development & quick time-to-market**
  - esp. important in telecom market

## cons

- **overhead** - ample memory & CPU power needed
  - more expensive card needed
- **complexity**
  - which brings security concerns
    - and abstraction can be a dangerous illusion for the defender

Financial sector much more conservative than telecom sector

# Multi-application cards

- multi-application vision: everyone carrying *one* card, with all their smartcard applications
- This is not going to happen. Problems include:
  - trust  
bank won't allow untrusted applet code on their cards, despite any VM+ firewall security guarantees
  - marketing  
who gets to put their logo on the plastic
- Still, multi-application is useful for development & card management
  - eg adding services to GSM SIM

# MULTOS

- card provides a Virtual Machine interpreting MEL (MULTOS Executable Language)
- originally developed for electronic purse system Mondex
  - by BT, Westminster & Midland banks in the UK
- designed for ITSEC EC6-high evaluation

# MULTOS: post-issuance application download

- MULTOS Certificate Authority (CA) key burnt into chip at manufacture
- Card-unique public/private key pair generated by CA; private key loaded directly after manufacture
- To load an application
  - CA creates Application Load Certificate (ALC)
    - permission to load application
    - ALC contains application ID, hash of code, and issuer public key
  - Issuer created Application Load Unit (ALU)
    - ALU unique for each card, so guaranteed to be loaded onto right card
  - Card loads ALC and ALU, verifies content, and installs

# JavaCard 3.0

- The next-generation smart card OS
  - multi-threading
    - security worries!
  - communication with https://
    - the smartcard is a web-server!
- But who will use it??
  - intended market: telco
  - not all card manufacturers produce JC 3.0, or have the intention to