Spelunking credit cards with Ruby

Chang Sau Sheong
8 Jan 2013
About me
About me
About me
About me
About me
About me
About me
About me
About me
About me
What computers are you carrying with you?
Smart card

A card with an embedded integrated circuit which has components for transmitting, storing and processing data
ID-1

ID-000 (mini SIM)

Mini-UICC (micro SIM)

nano SIM
Smart cards

Smart card

Processor card
- Can process data on card
- More memory storage
- More secured data
- 8 - 32-bit processors
- More expensive

Memory card
- Stores data only
- Performs fixed data manipulation
- Smaller memory storage
- Cheaper
History of smart cards

- 1968/69 - 2 German engineers Helmut Gröttrup and Jürgen Dethloff filed for patent for a chip on an ID card
- 1970 - similar patent filed by Kunitaka Arimura in Japan
- 1974 - Roland Moreno filed smart card patents in 11 countries
- 1977 - Michel Ugon from Honeywell Bull invented the first microprocessor smart card
- 1983/84 - First mass use of smart cards as telephone cards by French PTT
- 1991 - First SIM cards created by German smart card manufacturer G&D
- 1992 - smart cards used in Carte Bleue debit cards
- 1996 - EMV specification first published, version 3.1.1
- As of Q2 2012, there were 1.55 billion EMV compliant cards in use worldwide
- As end of 2011, there are about 6 billion GSM subscribers in the world
Standards

- Development and functionality of smart cards strongly driven by international standards
  - Interchangeability and interoperability very important
  - No particular vendor has dominant position
The International Electrotechnical Commission (IEC) and the International Organization for Standardization (ISO) are defined as follows: the IEC covers the fields of electrical technology and electronics, while ISO covers all other fields. Combined working groups are formed to deal with themes of common interest, and these groups produce combined ISO/IEC standards. Most standards for smart cards belong to this category.

ISO and the European standardization committee CEN (Comité Européen de Normalisation) also agree on rules for the development of standards that are recognized as both European and international standards. This leads to time and cost savings.

International standardization of smart cards

International standards for smart cards are developed under the auspices of ISO/IEC, and on the European level by the CEN. The major industrial countries are represented in all relevant committees, and they generally also maintain 'mirror' committees in the form of national working groups and voting committees. In Germany, this responsibility is borne by the DIN.

Figure 1.2 shows an overview of the structure of the relevant ISO and IEC working groups and the standards for which they are responsible.

As can be seen, there are two technical committees that are concerned with the standardization of smart cards. The first is ISO TC68/SC6, which is responsible for the standardization of cards used in the financial transaction area, while the second is ISO/IEC JTC1/SC17, which is responsible for general applications. This division has historical roots, since the first international applications were for identification cards used for financial transactions. The number of applications has naturally increased enormously since then, so the general standards, which are...
ISO/IEC 7816

- 7816-1
  - Physical characteristics of a card
  - For card manufacturers

- 7816-2
  - Dimension, location, functions of contacts
  - For card manufacturers

- 7816-3
  - Electronic signals, transmission protocols
  - For reader manufacturers

- 7816-4
  - Commands, messages, responses, files and data
  - For application developers

- 7816-5
  - Registration for application identifiers (AID)

- 7816-6
  - Inter industry data elements
Smart card OS

- Stored in the ROM of the microcontroller in unalterable form
- Classified into:
  - Native operating systems
    - OS and applications execute in machine language
  - Interpreter-based operating systems
    - OS in machine language, applications can be written in another language
    - Most popular include Java Card, MultOS and BasicCard
Application types

• Memory-based applications
  ‣ The terminal accesses the entire memory for read and write operations
  ‣ Can require certain conditions such as a PIN verification
  ‣ Limited in terms of their complexity, typical use include transit cards

• File-based applications
  ‣ Require processor cards and a smart card OS
  ‣ A set of data files (EFs) located in a directory file (DF)
  ‣ The smart card OS provides a large number of commands for data access, authentication and other operations

• Code-based applications
  ‣ Also use data files, but includes application-specific program code that can be executed in the smart card
  ‣ Examples include Java Card, BasicCard, Multos
File management

- Smart card file structures based on a tree structure with a root directory called MF (master file)
- The directories of a smart card are called DFs (dedicated files)
- The actual application data and operating system data are stored in EFs (elementary file)
Identifying files

- Standard filename consists of a 2-byte data element called the FID (file identifier). The FID of the MF is '3F00'.
- Each DF has a DF name in addition to its FID and includes an AID (application identifier).
  - The AID consists of an RID (registered application provider identifier) and a PIX (proprietary application identifier extension). RIDs can be registered officially to ensure that they are unique throughout the world.
- Each EF has an SFI (short file identifier) which can be provided as a parameter of a read or write command to select the EF directly.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>File Name</th>
<th>Size</th>
<th>Value Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>MF (master file)</td>
<td>FID (file identifier)</td>
<td>2 bytes</td>
<td>'3F00'</td>
</tr>
<tr>
<td>DF (dedicated file)</td>
<td>FID (file identifier)</td>
<td>2 bytes</td>
<td>0 … 'FFFD'</td>
</tr>
<tr>
<td></td>
<td>DF name (usually includes an AID)</td>
<td>1–16 bytes</td>
<td>0 … 'F … F'</td>
</tr>
<tr>
<td></td>
<td>AID (RID</td>
<td></td>
<td>PIX)</td>
</tr>
<tr>
<td>EF (elementary file)</td>
<td>FID (file identifier)</td>
<td>2 bytes</td>
<td>0 … 'FFFD'</td>
</tr>
<tr>
<td></td>
<td>SFI (short file identifier)</td>
<td>5 bits</td>
<td>1 … '30'</td>
</tr>
</tbody>
</table>
Interfacing with smart cards

- Communication with contact smart cards takes place via a half-duplex, bit-serial link
- This means that only one of the communicating parties can transmit at any given time
- To prevent collisions, it is necessary to fix which party initiates communication
- For smart cards, the terminal always initiates communications, which means it is the master and the smart card is the slave

  - This means the smart card transmits data only in response to a request from the terminal.

- This master–slave principle pervades all communications with smart cards
Figure 6.1: The initial data transfers between a terminal and a smart card, showing the answer to reset (ATR), protocol parameter selection (PPS) and the first command–response pair.

The entire procedure for data transmission to and from the smart card can be represented using the OSI layer model. This differentiates electrical events on the I/O line, logical processes in the actual transmission protocol and the behavior of applications that use these processes. The behavior and interactions within and between these layers are specified in several international standards. These relationships are illustrated in Figure 6.3.

In this chapter, the asynchronous transmission protocols are described with respect to relevant standards in terms of their functions. All allowed parameters and settings within the context of the protocol are described. In practice, it often happens that smart cards do not support all options of the transmission protocol, due to the limitations of available memory.

*Protocol Parameter Selection*

Sunday, 7 July, 13
The initial data transfers between a terminal and a smart card, showing the answer to reset (ATR), protocol parameter selection (PPS) and the first command–response pair.

Smart Card

Reset

Terminal

PPS request

PPS response

command 1

response 1

...
Smart Card Data Transmission

Figure 6.1 The initial data transfers between a terminal and a smart card, showing the answer to reset (ATR), protocol parameter selection (PPS) and the first command–response pair.

Figure 6.2 The general states of a smart card for activation and communication with the terminal.

The entire procedure for data transmission to and from the smart card can be represented using the OSI layer model. This differentiates electrical events on the I/O line, logical processes in the actual transmission protocol and the behavior of applications that use these processes. The behavior and interactions within and between these layers are specified in several international standards. These relationships are illustrated in Figure 6.3.

In this chapter, the asynchronous transmission protocols are described with respect to relevant standards in terms of their functions. All allowed parameters and settings within the context of the protocol are described. In practice, it often happens that smart cards do not support all options of the transmission protocol, due to the limitations of available memory.

*Protocol Parameter Selection*
Figure 6.1 The initial data transfers between a terminal and a smart card, showing the answer to reset (ATR), protocol parameter selection (PPS) and the first command–response pair.

Figure 6.2 The general states of a smart card for activation and communication with the terminal.

The entire procedure for data transmission to and from the smart card can be represented using the OSI layer model. This differentiates electrical events on the I/O line, logical processes in the actual transmission protocol and the behavior of applications that use these processes. The behavior and interactions within and between these layers are specified in several international standards. These relationships are illustrated in Figure 6.3.

In this chapter, the asynchronous transmission protocols are described with respect to relevant standards in terms of their functions. All allowed parameters and settings within the context of the protocol are described. In practice, it often happens that smart cards do not support all options of the transmission protocol, due to the limitations of available memory.

*Protocol Parameter Selection*
Figure 6.1 The initial data transfers between a terminal and a smart card, showing the answer to reset (ATR), protocol parameter selection (PPS) and the first command–response pair.

Figure 6.2 The general states of a smart card for activation and communication with the terminal.

The entire procedure for data transmission to and from the smart card can be represented using the OSI layer model. This differentiates electrical events on the I/O line, logical processes in the actual transmission protocol and the behavior of applications that use these processes. The behavior and interactions within and between these layers are specified in several international standards. These relationships are illustrated in Figure 6.3.

In this chapter, the asynchronous transmission protocols are described with respect to relevant standards in terms of their functions. All allowed parameters and settings within the context of the protocol are described. In practice, it often happens that smart cards do not support all options of the transmission protocol, due to the limitations of available memory.
The initial data transfers between a terminal and a smart card, showing the answer to reset (ATR), protocol parameter selection (PPS) and the first command–response pair:

- **Smart Card**
  - Smart card in sleep mode
  - Send ATR
  - Smart card electrically activated
  - Activation sequence

- **Terminal**
  - PPS request
  - PPS response
  - PPS necessary?
  - Yes
  - Command 1
  - Response 1
  - ...
Figure 6.1 The initial data transfers between a terminal and a smart card, showing the answer to reset (ATR), protocol parameter selection (PPS) and the first command–response pair.

- Smart Card
  - Reset
  - ATR
  - PPS request
  - PPS response
  - command 1
  - response 1
  - ...

- Terminal
  - PPS necessary?
    - yes
    - no

*Protocol Parameter Selection*
Communicating with smart cards

• The T=0 transmission protocol
  ‣ The oldest and most widely used protocol for smart cards including SIM cards
  ‣ Byte-oriented transmission protocol with relatively poor layer separation so Case 4 commands are not possible
  ‣ Terminal must use the GET RESPONSE command to retrieve data to be provided

• The T=0 transmission protocol
  ‣ Block-oriented T=1 protocol has distinct layer separation, so all four cases of command APDUs can be used
  ‣ Has a significantly more complicated structure than T=0, but it is also more robust
  ‣ Often used with payment cards and ID cards
Message structure

• Applications protocol data units (APDUs) are used to exchange all data that passes between the smart card and the terminal
• Holds a complete command to the card or a complete response from the card
• APDU commands sent to card are called command APDUs
• APDU response sent to terminal are called response APDUs
### 2.3 Data Transmission

<table>
<thead>
<tr>
<th>Case4 command APDU</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case3 command APDU</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case2 command APDU</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case1 command APDU</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response APDU, variant 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response APDU, variant 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW1</td>
</tr>
</tbody>
</table>

---

#### 2.3.3.1 T=0 transmission protocol for contact cards

The T=0 transmission protocol is the oldest and most widely used protocol for smart cards. It is a byte-oriented transmission protocol with relatively poor layer separation. As a result, Case 4 commands are not possible with T=0. Instead, the terminal must use the GET RESPONSE command to retrieve data to be provided to the terminal by the smart card. However, this has not significantly restricted the use of the T=0 protocol, which is the standard protocol for the world's largest smart card application: the SIMs and USIMs used in GSM and UMTS mobile telecommunication systems.

#### 2.3.3.2 T=1 transmission protocol for contact cards

The block-oriented T=1 protocol has distinct layer separation, so all four cases of command APDUs can be used with this protocol. T=1 has a significantly more complicated structure than T=0, but it is also significantly more robust, thanks to its processes for detecting and resending blocks that contain transmission errors. T=1 is often used with payment cards and ID cards. It is indisputably a more modern protocol than T=0, but its advantages relative to T=0 are not large enough to threaten T=0 with becoming irrelevant.

#### 2.3.3.3 USB transmission protocol for contact cards

The data transmission rate of T=0 or T=1 rarely exceeds 115 kbps in practice. This is too low for smart cards with large data memories. This is one of the reasons why the USB protocol (Universal Serial Bus) is slowly becoming established in the smart card world. The second main reason is that USB provides compatibility with the PC environment. USB smart cards that support the 1.5 Mbps data rate of low-speed USB and even the 480 Mbps data rate of high-speed USB.
Command APDUs

- **CLA** - class byte
- **INS** - instruction byte
- **P1, P2** - parameter bytes
- **Lc** - length of command data
- **Le** - length of expected response data
- **Example command APDUs**
  - SELECT FILE
  - READ RECORD
  - GET RESPONSE
Response APDUs

- SW1, SW2 - status word 1 and 2
- Example SW1, SW2:
  - Normal response
    - 90 00 - Ok
    - 61 xx - Has more data, length of data is SW2
  - Warning response
    - 62 81 - return data corrupted
    - 63 00 - authentication failed
  - Error response
    - 68 00 - request not supported
    - 6A 82 - File not found
PC communications

- PC/SC (Personal Computer/Smart Card)
  - De facto specification for smart card integration with PCs
  - Default in Windows, ported to Linux with PC/SC Lite, forked version in OS X
- CT-API (Card Terminal API)
  - Alternative, older specification
  - Single application, single user
- OpenCT
  - Alternative open source driver
  - Not standard
EMV

- EMV (Europay, Mastercard, Visa) is a global standard for credit and debit payment cards based on chip card technology
- First published 1996 version 3.1.1
- Current version 4.3 November 2011
- JCB joined 2004, American Express joined 2009
- Controlled by EMVCo, with 25% shareholdings amongst Visa, Mastercard, American Express and JCB
- Defines interaction at physical, electrical, data and application layers between smart card and terminals for financial transactions
EMV

- Standards based on ISO/IEC 7816 for contact cards, ISO/IEC 14443 for contactless cards
- As of Q2 2012, there were 1.5 billion EMV compliant cards in use worldwide
- Main purposes for increased security (reducing fraud) and finer control of offline transactions
- Multiple implementations of EMV -
  - VSDC - Visa
  - M/Chip - Mastercard
  - AEIPS - American Express
  - J Smart - JCB
  - D-PAS - Discover/Diners Club International
# EMV Adoption

**Region** | **EMV Cards** | **Adoption Rate** | **EMV Terminals** | **Adoption Rate**
--- | --- | --- | --- | ---
Canada, Latin America, and the Caribbean | 318,779,062 | 41.1% | 4,443,000 | 76.7%
Asia Pacific | 366,229,237 | 28.2% | 4,551,000 | 51.4%
Africa & the Middle East | 31,573,578 | 20.6% | 462,000 | 75.9%
Europe Zone 1 | 759,760,119 | 84.4% | 11,920,000 | 94.4%
Europe Zone 2 | 37,104,467 | 14.5% | 610,500 | 68.1%
United States† | | | | |
TOTALS | 1,513,446,463 | 44.7% | 21,986,500 | 76.4%

*Figures reported in Q4 2011 and represent the latest statistics from American Express, JCB, MasterCard and Visa, as reported by their member financial institutions globally.

† Figures do not include data from the United States.

- Figures as of end 2011
- Does not include US (slow adoption - cost, weak justification, large number of banks)
- Liability shift to acquirers over next 3 (Visa) - 5 (Mastercard) years in US
(typical) EMV flow

- Detect card and reset
- List applications
- Select application
- Get data
- Authenticate data
- Verify cardholder
- Find processing restrictions
- Manage risk
- Terminal decides action
- Card decides action (card wins)
- Process online/offline
- Card decides after processing
- Transaction complete

Sunday, 7 July, 13
TLV

- Tag-Length-Value
- Tag - 1 or 2 bytes
- Length - length of the value (in bytes)
- Value - actual data
- Can be nested or in sequence
Let's dive in.
Detect card and reset

• Insert card
• Wait for ATR
• Show ATR
Payment System Environment

- PSE is a DDF with the name 1.PAY.SYS.DDF01
- Contains one or more EMV applications
- Doesn’t always exist
List applications

• SELECT the PSE

• If PSE doesn’t exist, go through list of AIDs that the terminal supports to get the list of EMV applications

• If PSE exists, use GET_RESPONSE to get the PSE FCI

• PSE FCI has the SFI to the PSE record

• Use GET_RECORD with SFI to get the PSE record

• PSE record has ADFs of the EMV applications
11.2 READ RECORD Command-Response APDUs

Application Independent ICC to Terminal Interface Requirements

11.2.2 Command Message

The READ RECORD command message is coded according to Table 38:

<table>
<thead>
<tr>
<th>CLA</th>
<th>INS</th>
<th>P1 (record number)</th>
<th>P2 (reference control parameter)</th>
<th>Lc</th>
<th>Data</th>
<th>Le</th>
</tr>
</thead>
<tbody>
<tr>
<td>'00'</td>
<td>'B2'</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 38: READ RECORD Command Message

Table 39 defines the reference control parameter of the command message:

<table>
<thead>
<tr>
<th>b8</th>
<th>b7</th>
<th>b6</th>
<th>b5</th>
<th>b4</th>
<th>b3</th>
<th>b2</th>
<th>b1</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td>SFI</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 39: READ RECORD Command Reference Control Parameter

11.2.3 Data Field Sent in the Command Message

The data field of the command message is not present.

11.2.4 Data Field Returned in the Response Message

The data field of the response message of any successful READ RECORD command contains the record read. Records read during application selection are directory records which are formatted as in section 12.2.3. The format of records read during application processing is application dependent.

11.2.5 Processing State Returned in the Response Message

'9000' indicates a successful execution of the command.
Select application

- ADF represents 1 EMV application
- SELECT the ADF to get the ADF FCI
- ADF FCI has information on application including the PDOL (Processing Options Data Object List)
  - PDOL tells the terminal what the card needs
  - PDOL doesn’t always exist, if there is no PDOL use 83 00
- use GET_PROCESSING_OPTIONS with the PDOL to initiate the EMV transaction
Get data from card

- GPO returns the AIP (Application Interchange Profile) and AFL (Application File Locator)
- AIP tells the terminal which features are supported
- AFL tells the terminal while files and records can be read
AIP

AIP tells the terminal:

- What features are supported by the application
- Whether terminal risk management should be performed

### AIP Byte 1 (Leftmost)

<table>
<thead>
<tr>
<th>b8</th>
<th>b7</th>
<th>b6</th>
<th>b5</th>
<th>b4</th>
<th>b3</th>
<th>b2</th>
<th>b1</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>RFU</td>
</tr>
<tr>
<td>x</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>SDA supported</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>DDA supported</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>Cardholder verification is supported</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>Terminal risk management is to be performed</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>Issuer authentication is supported</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>RFU</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>CDA supported</td>
</tr>
</tbody>
</table>

### AIP Byte 2 (Rightmost)

<table>
<thead>
<tr>
<th>b8</th>
<th>b7</th>
<th>b6</th>
<th>b5</th>
<th>b4</th>
<th>b3</th>
<th>b2</th>
<th>b1</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>RFU</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>RFU</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>RFU</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>RFU</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>RFU</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>RFU</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>RFU</td>
</tr>
</tbody>
</table>
That’s it folks
(for now)