

Travaux pratiques

Protocoles ISO 7816-3, ISO 7816-4

Standards

TP n°1 – Interaction avec la carte SIM - Les premières commandes APDUs en utilisant un outil de script PC/SC

TP n°2 – Interaction avec la carte EMV

TP n°3 – Programmation d'une application Java pour explorer la carte SIM (TP BONUS)

Rappels

1. Introduction

Les normes ISO 7816 « Identification cards – Integrated circuit cards with contacts » ont été publiées par l'organisation internationale de normalisation (ISO, International Organisation for Standardisation). C'est le plus important standard définissant les caractéristiques des cartes à puce qui fonctionnent avec un contact électrique. Sachant que 15 normes sont proposées pour les cartes à contact, nous décrivons brièvement ici uniquement les 7 premières normes.

1.1 ISO 7816-1

Cette norme définit les caractéristiques physiques des cartes à puce à contact : la géométrie, la résistance, les contacts, etc.

1.2 ISO 7816-2

Cette norme spécifie le dimensionnement physique (extérieur) des contacts de la puce. Deux des huit contacts réservés à une utilisation future (RFU) sont redéfinis pour l'utilisation USB dans la norme ISO 7816-12.

1.3 ISO 7816-3

Cette norme définit l'interface électrique et les protocoles de transmission :

- les protocoles de transmission (TPDU, Transmission Protocol Data Unit) : T=0 : protocole orienté octet, T1 : protocole orienté paquet, T=14 : réservé pour les protocoles propriétaires.
- la sélection d'un type de protocole.
- la réponse à un reset (ATR, ou Answer To Reset en anglais) qui correspond aux données envoyées par la carte immédiatement après la mise sous tension.
- les signaux électriques, tels que le voltage, la fréquence d'horloge et la vitesse de communication.

1.4 ISO 7816-4

Cette norme vise à assurer l'interopérabilité des échanges. Elle définit les messages APDU (Application Protocol Data Units), par lesquels les cartes à puce communiquent avec le lecteur. Les échanges s'effectuent en mode client-serveur, le terminal ayant toujours l'initiative de communication.

1.5 ISO 7816-5

Cette norme définit le système de numérotation et les procédures d'enregistrement et d'attribution des identifiants des applications (AID, ou Application IDentifier). Un unique AID est associé à chaque application et à certains fichiers sur la carte. Ils sont représentés par des tableaux d'octets de taille allant de 5 à 16. Les cinq premiers octets représentent le numéro d'enregistrement du fournisseur d'application (RID, Registered Application Provider IDentifier en anglais) qui est attribué par la Copenhagen Telephone Company Ltd ou l'ISO. Ils sont suivis par l'identifiant

optionnel PIX (Proprietary Application Identifier eXtension) d'une longueur allant jusqu'à 11 octets.

L'identifiant RID est le même pour le paquetage et l'applet, mais le PIX doit être différent.

1.6 ISO 7816-6

Cette norme spécifie des éléments de données inter-industrie pour les échanges, tels que le numéro du porteur de carte, sa photo, sa langue, la date d'expiration, etc.

1.7 ISO 7816-7

Cette norme définit les commandes inter-industrie pour langage d'interrogation de carte structurée (SCQL).

2. ATR (Answer To Reset) défini dans l'ISO 7816-3

Dès que la carte est mise sous tension, elle envoie un message de réponse d'initialisation appelé ATR, il peut atteindre une taille maximale de 33 octets. Il indique à l'application cliente les paramètres nécessaires pour établir une communication avec elle. Il fournit un nombre varié de paramètres liés à la carte et au protocole de transmission utilisé :

- Le protocole de transport ;
- Taux de transmission des données ;
- Numéro de série de la puce ...

Le premier octet noté TS ="3F" pour convention indirecte ou "3B" pour convention directe.

3. Échange de commandes avec le lecteur de carte à puce tel que défini dans l'ISO 7816-4

La communication entre l'hôte et la carte est half-duplex. Elle se fait à l'aide de paquets appelés APDU (Application Protocol Data Units) en respectant le protocole de l'ISO 7816-4. Un APDU contient une commande ou une réponse. Le mode Maître/Esclave est utilisé. Ainsi la carte joue un rôle passif et attend une commande APDU à partir de l'hôte. Elle exécute l'instruction spécifiée dans la commande et retourne une réponse APDU.

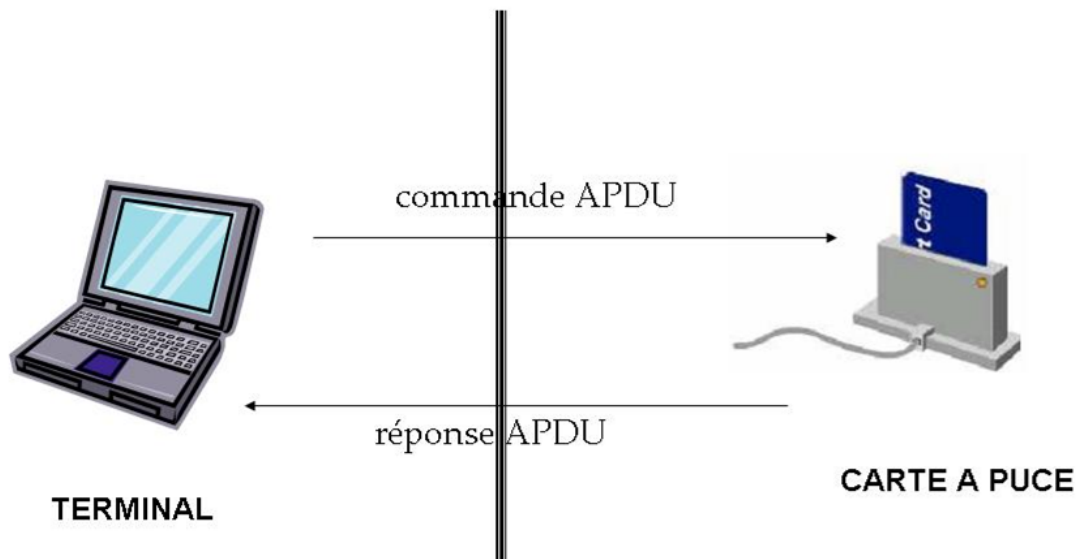


Figure 1 : Le modèle de communication de la carte à puce

3.1. Format des commandes APDU

Commande APDU						
Entête obligatoire				Corps optionnel		
CLA	INS	P1	P2	Lc	Data field	Le
➤ CLA (1 octet) : Classe d'instructions : indique la structure et le format pour une catégorie de commandes et de réponses APDU. ➤ INS (1 octet) : code d'instruction : spécifie l'instruction de la commande. ➤ P1 (1 octet) et P2 (1 octet) : paramètres de l'instruction. ➤ Lc (1 octet) : nombre d'octets présents dans le champ données de la commande. ➤ Data field (octets dont le nombre est égal à la valeur de Lc) : une séquence d'octets dans le champ données de la commande.						

3.2. Format des réponses APDU

Réponse APDU		
Corps optionnel		Partie obligatoire
Data field		SW1 SW2
➤ Data field (longueur variable) : une séquence d'octets reçus dans le champ données de la réponse. ➤ SW1 (1 octet) et SW2 (1 octet) : Status words (Mots d'état) état de traitement par la carte.		

4. Exemples de cartes

Le tableau suivant donne des exemples de commandes APDU utilisées dans le monde de la carte.

Champ de la commande APDU	Valeurs
CLA	BC = cartes de crédit françaises, cartes vitales françaises, A0 = cartes SIM (téléphonie) 00 = cartes Monéo (porte-monnaie en France), Mastercard, Visa
INS	20 = présentation du PIN, 40 = validation (ratification du code PIN) B0 = Lecture B2 = Lecture de record D0 = Écriture DC = Écriture de record A4 = Sélection du répertoire (directory) C0 = Demander une réponse (get response)
P1, P2	paramètres contenant des adresses à lire
LEN	longueur prévue pour la réponse ou bien longueur de l'argument de l'instruction.
ARG	contient LEN octets (octets à écrire, PIN à vérifier, etc.)

La réponse APDU sert à accuser réception la commande APDU envoyée par le terminal. Ainsi, la carte répond en envoyant le code instruction INS, suivi de données de longueur LEN en terminant par SW1 et SW2 (0x90 0x00 lorsque la commande s'est déroulée avec succès). En cas d'échec, seuls les champs SW1 et SW2 seront envoyés au terminal avec les codes d'erreur suivants :

0x6E 0x00	CLA error
0x6D 0x00	INS error
0x6B 0x00	P1, P2 error
0x67 0x00	LEN error
0x98 0x04	Bad PIN
0x98 0x08	Unauthorized Access
0x98 0x40	Card blocked
...	

TP n°1 – Interaction avec la carte SIM
Les premières commandes APDUs en utilisant un outil de script PC/SC

Objectif : Manipulation d'outils de communication avec une carte à puce et appropriation d'une spécification SIM (exploration de l'arborescence).

Vous allez explorer l'arborescence de fichiers d'une carte à puce. Nous avons choisi d'utiliser une carte SIM car son arborescence de fichiers est très riche. D'autre part, les informations à lire ne sont pas toutes confidentielles. On utilisera la spécification 3GPP TS 11.11 fournie.

Sous l'environnement Windows : Télécharger le logiciel TestResMan

Le téléphone portable, dès qu'il est mis sous tension, sélectionne le répertoire GSM, détecte si la présentation d'un code PIN est nécessaire, et présente cette valeur via la commande VERIFY. Ensuite, le portable lit le fichier EF-Phase (FID=6FAE) qui contient le numéro de version fonctionnel de la carte. Après cela, le téléphone peut écrire ou lire différents fichiers. C'est ce que nous allons faire dans cet exercice.

Réaliser la suite des commandes suivantes en utilisant le tableau des commandes APDU donné ci-dessous.

- Sélection du répertoire GSM
- Présentation du code PIN
- Lecture de l'IMSI
- Lecture des paramètres TMSI et LAI
- Exécution de l'algorithme d'authentification du GSM
- Mise à jour du fichier EF-Kc
- Lecture de la table des services SIM (EF-SIM-Service-Table)
- Lecture et écriture des SMS dans la SIM
- Lecture de l'annuaire des numéros, ADN

COMMANDE	INS	P1	P2	P3
<i>SELECT</i>	A4	00	00	02
<i>STATUS</i>	F2	00	00	Length
<i>READ BINARY</i>	B0	Offset high	Offset low	Length
<i>UPDATE BINARY</i>	D6	Offset high	Offset low	Length
<i>READ RECORD</i>	B2	Record number	Mode	Length
<i>UPDATE RECORD</i>	DC	Record number	Mode	Length
<i>SEEK</i>	A2	00	Type/mode	Length
<i>INCREASE</i>	32	00	00	03
<i>VERIFY CHV</i>	20	00	CHV number	08
<i>CHANGE CHV</i>	24	00	CHV number	10
<i>DISABLE CHV</i>	26	00	01	08
<i>ENABLE CHV</i>	28	00	01	08
<i>UNBLOCK CHV</i>	2C	00		10
<i>INVALIDATE</i>	04	00	00	00
<i>REHABILITATE</i>	44	00	00	00
<i>RUN GSM ALGORITHM</i>	88	00	00	10
<i>SLEEP</i>	FA	00	00	00
<i>GET RESPONSE</i>	C0	00	00	Length
<i>TERMINAL PROFILE</i>	10	00	00	Length
<i>ENVELOPE</i>	C2	00	00	Length
<i>FETCH</i>	12	00	00	Length
<i>TERMINAL RESPONSE</i>	14	00	00	Length

TP n°2 – Interaction avec la carte EMV

Les premières commandes APDUs en utilisant un outil de script PC/SC

Objectif : Manipulation d'outils de communication avec une carte à puce et appropriation d'une spécification EMV.

Vous allez explorer votre carte bancaire à la recherche des informations suivantes : prénom, nom, numéro de la carte, etc.

Sachez bien qu'il n'y a aucun risque d'interroger votre carte bancaire (envoyer des commandes APDUs), car la plupart du temps, il s'agit juste de lecture mémoire, sauf bien entendu, pour quelques instructions telles que la ratification du code PIN (Personal Identification Number) qui s'il est saisi de façon erronée trois fois de suite risque de bloquer la carte.

Dans ce TP, nous explorons l'application bancaire EMV (et non plus, feu, l'application bancaire française B0'). Les spécifications sont publiques et téléchargeable ici : <http://www.emvco.com/specifications.aspx?id=223> (Il faut télécharger les Books 1 et 3 !)

Sous l'environnement Windows : Télécharger le logiciel TestResMan

- 1) Trouvez une facturette et chercher l'AID dessus ! Si vous n'en avez pas sélectionnez le PSE.
- 2) Sélectionnez l'application EMV.
- 3) Récupérez les données avec la commande GET_RESPONSE.

En effet, la prochaine étape après la sélection consistera à envoyer la commande "GET PROCESSING OPTIONS" (GPO) pour récupérer l'Application Interchange Profile (AIP) et l'Application File Locator (AFL). Or, pour envoyer la commande GPO, vous aurez besoin de la Processing Data Objects List (PDOL) pour le champ de données de la commande GPO. La PDOL est une partie de la réponse à la sélection de l'application comme vous pouvez le voir dans la spécification.

Tag	Value		Presence
'6F'	FCI Template		M
	'84'	DF Name	M
	'A5'	FCI Proprietary Template	M
	'50'	Application Label	M
	'87'	Application Priority Indicator	O
	'9F38'	PDOL	O
	'5F2D'	Language Preference	O
	'9F11'	Issuer Code Table Index	O
	'9F12'	Application Preferred Name	O
	'BF0C'	FCI Issuer Discretionary Data	O
	'9F4D'	Log Entry	O
	'XXXX' (Tag constructed according to Book 3, Annex B)	1 or more additional proprietary data elements from an application provider, issuer, or IC card supplier, or EMV-defined tags that are specifically allocated to 'BF0C'	O

Le « tag » de la PDOL est “9F38” et la PDOL est optionnelle. Peut-être que dans la réponse de votre carte, il n'y aura pas de PDOL. Ne vous inquiétez pas, cela est courant.

4) Envoyez la commande GPO et récupérez les données avec la commande GET_RESPONSE. Déterminez en fonction de ce que vous trouverez dans les spécifications l'AFL le ou les SFI (Short File Identifier) et les records associés dans lesquels vous pouvez lire des données.

5) Une fois les SFI et records déterminés, lisez les données avec la commande READ RECORD et essayez de trouver votre numéro de carte bancaire, votre nom et prénom et la date d'expiration.

6) Si vous avez fini sélectionnez le PSE et trouvez toutes les applications disponibles sur votre carte.

Vous trouverez des manipulations similaires à celles de ce TP sur ce blog : <http://blog.saush.com/2006/09/08/getting-information-from-an-emv-chip-card/>

Une fois le TP fini, si vous le souhaitez, vous pouvez utiliser CardPeek pour explorer votre carte :-) <http://pannetrat.com/Cardpeek/>

TP n°3 – Programmation d'une application Java pour explorer la carte SIM

Objectif : Être capable de développer ses propres applications pour communiquer avec une carte à puce.

Dans ce TP, nous utiliserons Eclipse et le package `javax.smartcardio` pour accéder à la carte depuis un programme Java.

Dans les propriétés du projet, il peut être nécessaire d'autoriser l'accès au package `javax.smartcardio`. Cela se traduit par l'ajout d'une règle d'accès dans la “Librairies” via ***Accessible javax/smartcardio/*****

Un programme minimum est :

```
import javax.smartcardio.*;

public class ClientPC {

    public static void main(String[] args) {

        // insérer le code ici !

    }

}
```