Compromission d'un terminal sécurisé via l'interface carte à puce

G. VINET

SERMA Technologies, CESTI

Symposium sur la sécurité des technologies de l'information et des communications, 2013

Logiciels couplés avec une carte à puce

Une sécurité basée sur un élément robuste

- La carte à puce : un excellent coffre-fort numérique.
- Une solution à faible coût et facile à déployer.
- Exemple : signature de document avec la carte d'identité électronique allemande.

PC, ton univers impitoyable

- Authentification via un code numérique fixe : le code PIN.
- Saisie du code PIN via le clavier du PC : non-sécurisé (CCC 2010).

Logiciels couplés avec une carte à puce

Une sécurité basée sur un élément robuste

- La carte à puce : un excellent coffre-fort numérique.
- Une solution à faible coût et facile à déployer.
- Exemple : signature de document avec la carte d'identité électronique allemande.

PC, ton univers impitoyable

- Authentification via un code numérique fixe : le code PIN.
- Saisie du code PIN via le clavier du PC : non-sécurisé (CCC 2010).

Outline

Terminal sécurisé

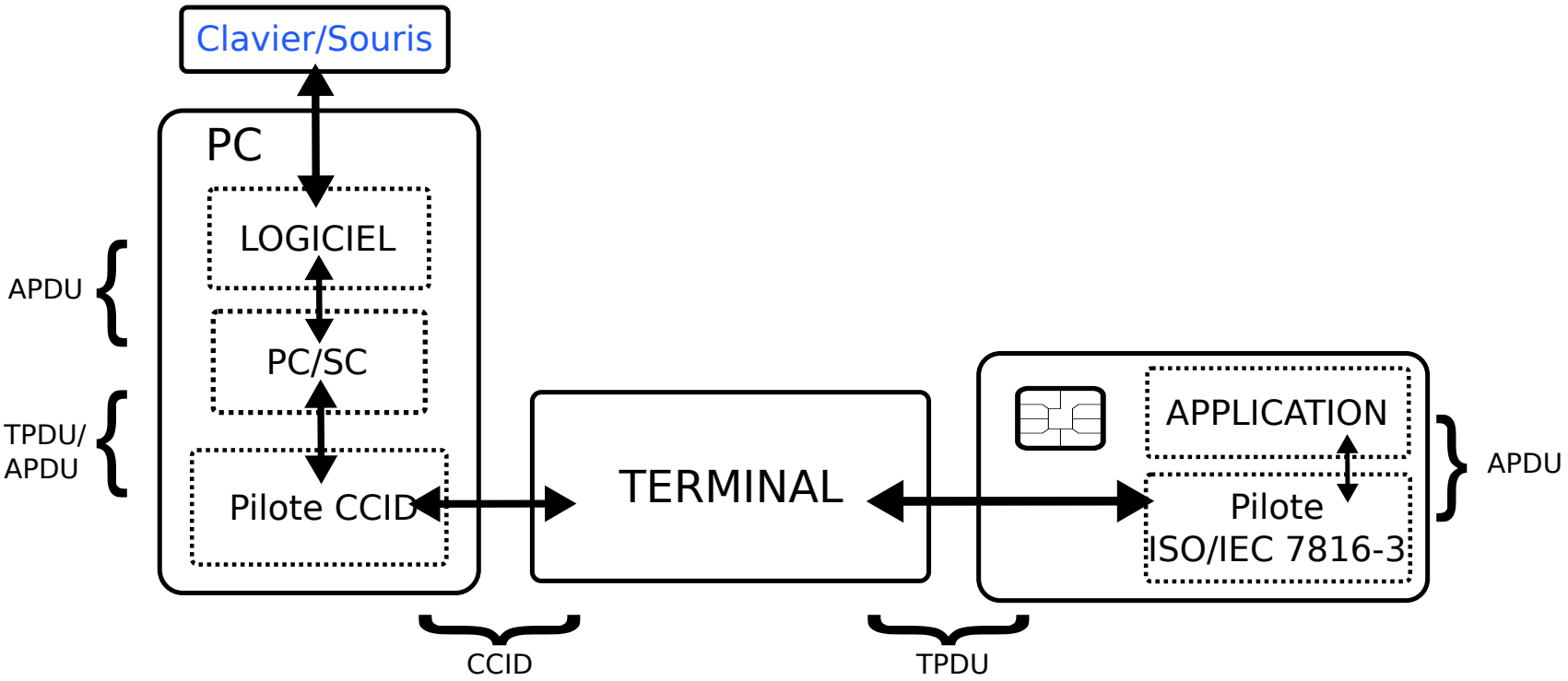
Apparence similaire à un terminal bancaire

- Affichage numérique sur deux lignes.
- Clavier numérique ainsi que touches de validation, d'annulation et de correction.
- Branchement en USB sur le PC.

Un simple lecteur avec deux modes de fonctionnement

- Un lecteur USB classique.
- Un terminal de saisie sécurisée.

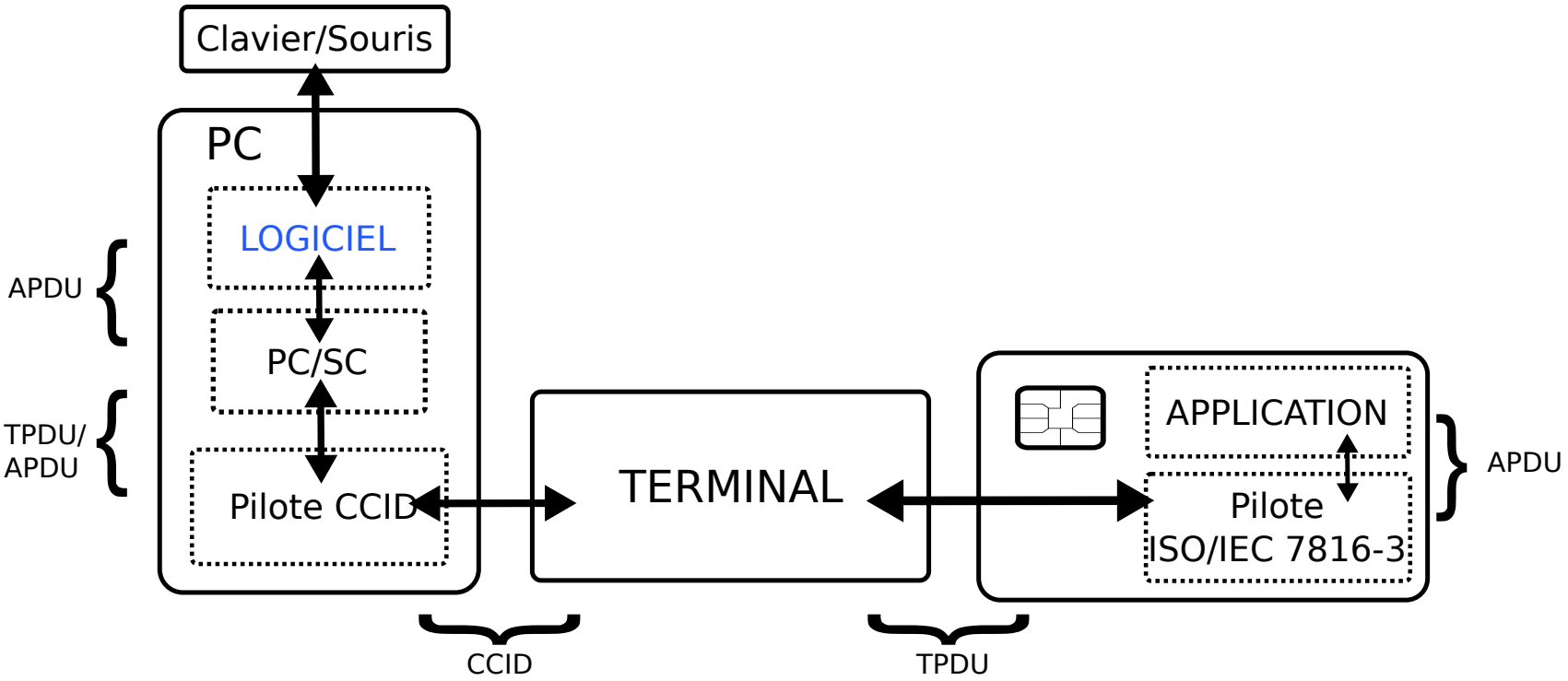
Un lecteur USB classique



Saisie utilisateur du code PIN

1234

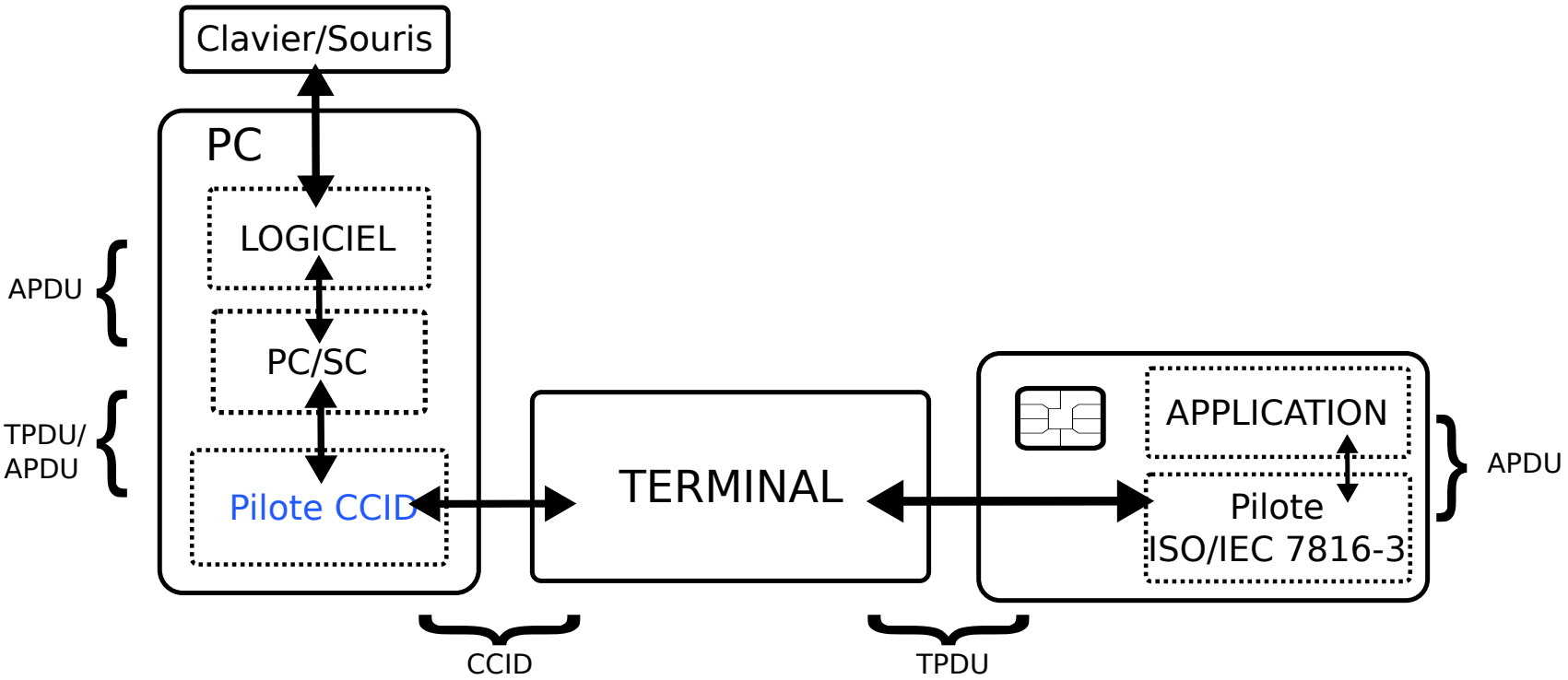
Un lecteur USB classique



Commande APDU (ISO 7816-4)

« Verify PIN » : 00 20 00 80 08 24 12 34 FF FF FF FF FF 00

Un lecteur USB classique

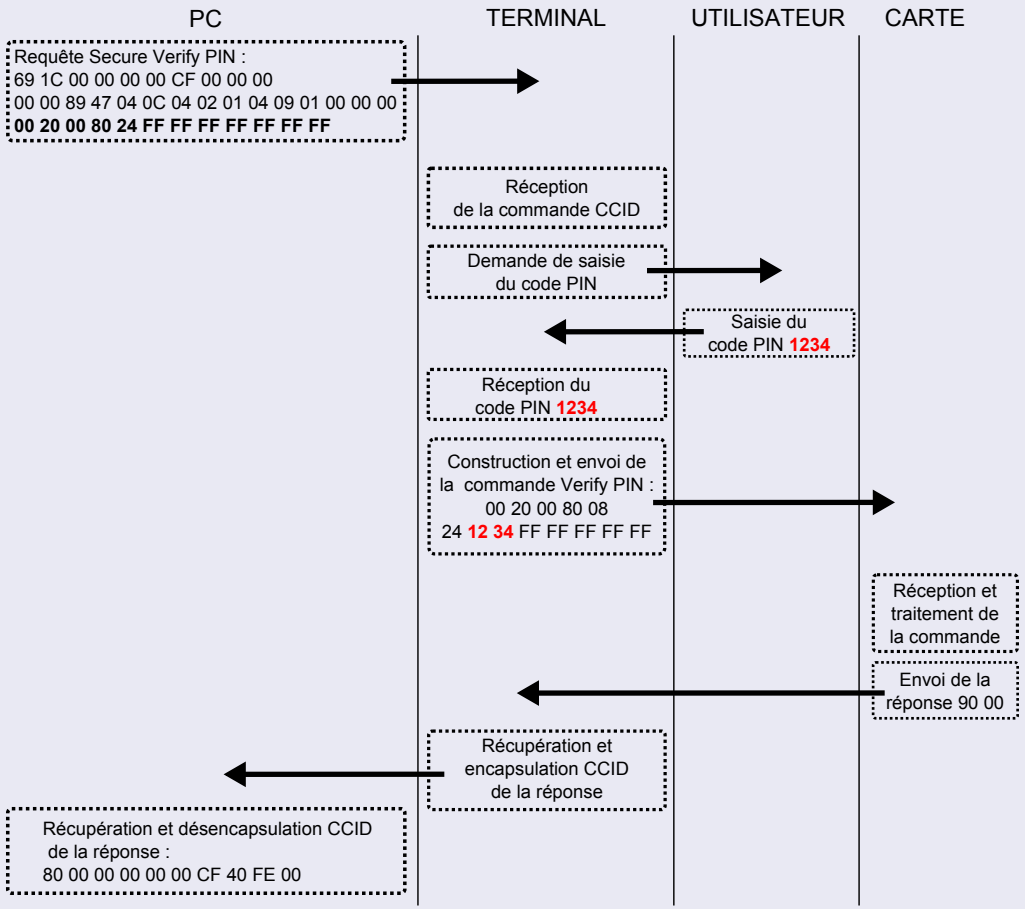


Commande CCID (Circuit(s) Cards Interfaces Devices)

6F 0E 00 00 00 00 19 00 00 00 00 20 00 80 08 24 12 34 FF FF FF FF FF 00

Un terminal de saisie sécurisé

PC/SC Part 10 : « IFDs with Secure PIN Entry Capabilities »



Outline

Modèle de sécurité

- Protection du PIN : cloisonnement au sein du terminal, hormis son envoi à la carte à puce.
- Pas de protection de l'interrogation de la carte une fois le PIN validé : problématique laissée à l'application de la carte.
  - Le terminal augmente la sécurité, mais ne l'assure pas à lui seul.
- Périmètre restreint et technologie à mettre en oeuvre rudimentaire : facile à sécuriser . . .vraiment ?

Outline

Attaque du terminal

Le but :

- Récupération du code PIN saisi sur le terminal lors d'une précédente session.
- Et éventuellement toutes autres données échangées...

Les moyens :

- Piégeage physique.
- Espionnage du rayonnement électromagnétique.
- Démontage et analyse des mémoires de masse ou temporaire.
- Attaque logicielle via l'interface clavier, USB . . . ou carte à puce.

Attaques via l'interface carte à puce

Pourquoi ?

- Rapide à mettre en oeuvre.
- Non destructif, donc difficilement détectable.
- Rarement pris en compte.

Mise en oeuvre

Carte esclave du terminal : besoin de stimuler son action malveillante.

- Utilisation du PC où le terminal est branché si disponible.
- Sinon « warm replug attack » : débrancher le terminal tout en le gardant sous tension pour ensuite le brancher sur un PC d'analyse.

Attaques via l'interface carte à puce

Pourquoi ?

- Rapide à mettre en oeuvre.
- Non destructif, donc difficilement détectable.
- Rarement pris en compte.

Mise en oeuvre

Carte esclave du terminal : besoin de stimuler son action malveillante.

- Utilisation du PC où le terminal est branché si disponible.
- Sinon « warm replug attack » : débrancher le terminal tout en le gardant sous tension pour ensuite le brancher sur un PC d'analyse.

Attaques applicatives inopérantes

Terminal transparent par rapport aux applications embarquées sur la carte :

- Pas d'analyse des réponses de la carte : décodage ou extraction de champs, calculs cryptographiques ...
- Transmission directe des réponses au PC, qui va les interpréter.

Attaque du protocole de communication bas niveau ISO 7816-3

Outline

Attaque du protocole ISO 7816-3

Transmission de la structure de base

- Application Protocol Data Unit (APDU)

Commande APDU

Entête	Corps de données	Le
00 20 00 80 08	24 12 34 FF FF FF FF FF	00
CLA INS P1 P2 Lc		

Réponse APDU

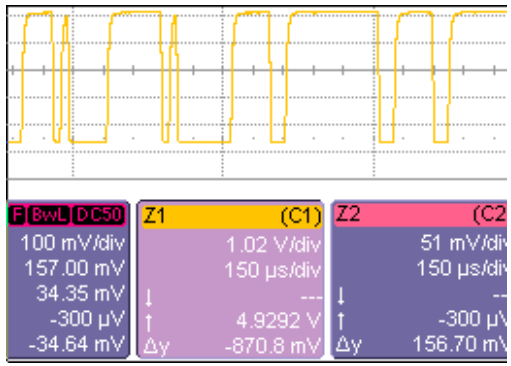
Corps de données	SW
87 48 64 78 45	90 00
	SW1 - SW2

Attaque du protocole ISO 7816-3

- Réduction de la surface d'attaque.
- Mais applicable à tout terminal communiquant en contact, quelque soit l'application.
- Norme ancienne, donc implémentation bien maîtrisée.
- Mais, encore complexe sur certains points.

L'Answer To Reset (ATR)

- 1ère réponse de la carte lors de sa mise sous tension.
- Codage des paramètres de la communication.
- Structure de taille variable, mais limitée à 33 octets.



2er octets obligatoires

- **TS** : codage d'un caractère sur la ligne physique.
- **T0** : indique le nombre d'octets historiques ou de configuration dont les tailles peuvent varier.

L'Answer To Reset (ATR)

Octets historiques

- Situés à la fin de l'ATR.
- Taille codée par le quartet de poids faible de **T0**, donc de 0 à 15 octets.

ATR	3B 03 45 78 68
3B	codage des octets en convention directe
03	l'ATR contient 3 octets historiques
45 78 68	octets historiques

L'Answer To Reset (ATR)

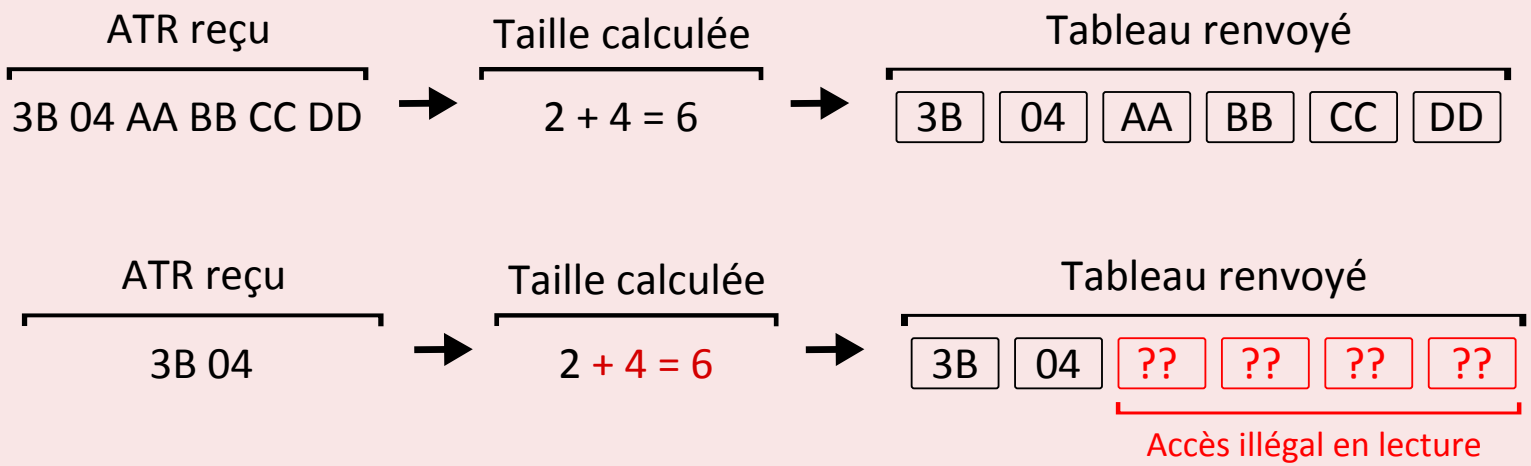
Octets de configuration

- Chaque bit du quartet de poids fort de **T0** indique la présence ou non des octets de configuration TA1, TB1, TC1, TD1,
- Même principe pour l'octet TD1 indiquant la présence des octets TA2 à TD2. Et ainsi de suite, ...

ATR	3B 13 96 45 78 68
3B	codage des octets en convention directe
13	1 octet de configuration et 3 octets historiques.
96	TA1
45 78 68	octets historiques

L'Answer To Reset (ATR)

Incohérence longueur reçue/longueur calculée



L'Answer To Reset (ATR)

Taille supérieure à 33 octets

ATR reçu

3B F7 00 00 00 F0 00 00
00 F0 00 00 00 F0 00 00
00 F0 00 00 00 F0 00 00
00 00 00 11 22 33 44 55
66

→ Taille calculée

2 + 4 + 4 + 4 +
4 + 4 + 4 + 7
= 33

→ Tableau modifié

3B	F7	00	00	00	F0	00	00
00	F0	00	00	00	F0	00	00
00	F0	00	00	00	F0	00	00
00	00	00	11	22	33	44	55
66	A2	05	04	18	B5	89	45
A5	74	38	49	BC	AD	42	00

ATR reçu

3B FF 00 00 00 F0 00 00
00 F0 00 00 00 F0 00 00
00 F0 00 00 00 F0 00 00
00 00 00 11 22 33 44 55
66 77 88 99 00 11 22 33
44

→ Taille calculée

2 + 4 + 4 + 4 +
4 + 4 + 4 + 15
= 41 > 33

→ Tableau modifié

3B	F7	00	00	00	F0	00	00
00	F0	00	00	00	F0	00	00
00	F0	00	00	00	F0	00	00
00	00	00	11	22	33	44	55
66	77	88	99	00	11	22	33
44	74	38	49	BC	AD	42	00

Accès illégal en écriture

Deux modes de transport pour l'APDU

- Mode T0 ou mode T1.
- Choix lors de la négociation PPS : « Protocol and Parameters Selection ».
- Impact sur :
  - la transformation de l'APDU en TPDU (Transport Protocol Data Unit).
  - l'envoi et l'acquittement de la TPDU entre le terminal et la carte.

Mode T0

- Communication orientée par octets.
- APDU et TPDU quasiment identiques.
- Envoi de la TPDU extrêmement segmenté par une machine à état pilotée par des « octets de procédures ».

Octets de procédure du mode T0

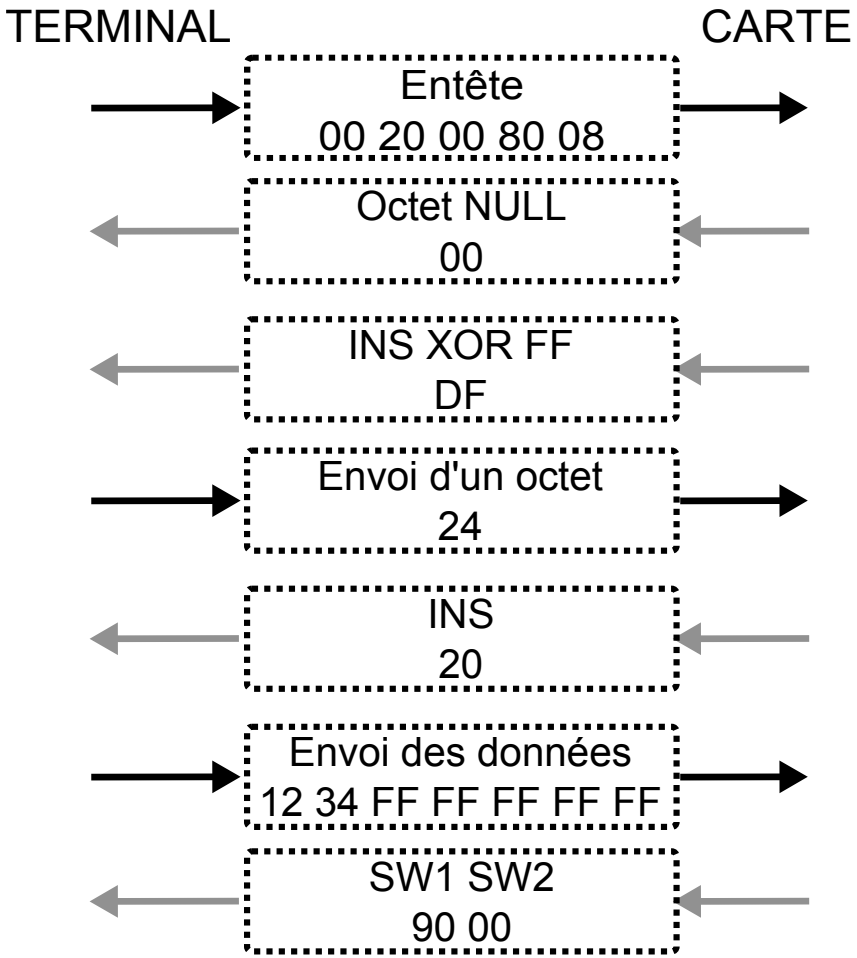
Exemple d'envoi d'un Verify PIN

Commande APDU

Entête	Corps de données	Le
00 20 00 80 08	24 12 34 FF FF FF FF FF	00
CLA INS P1 P2 Lc		

Réponse APDU

Corps de données	SW
87 48 64 78 45	90 00
	SW1 - SW2



Octets de procédure du mode T0

Mauvaise gestion des octets de procédure

Envoi entête

00 20 00 80 08

Etat interne du terminal

recLen = 0x00

recArray[256]

0x0000	FF	FF	FF	FF	FF	FF	FF	FF
	[...]							
0x00F0	FF	FF	FF	FF	FF	FF	FF	FF
0x0100	A5	FC	ED	12	58	8F	81	22

Réception 300 octets NULL

00 00 ... 00 00

Etat interne du terminal

recLen = 0x12C

recArray[256]

0x0000	00	00	00	00	00	00	00	00
	[...]							
0x00F0	00	00	00	00	00	00	00	00
0x0100	00	00	00	00	00	00	00	00

Accès illégal en écriture

Les APDU étendues

- Longueur du corps de données de l'APDU codée sur 1 ou 2 octets.
- Taille maximale de 256 ou 65536 octets. Dans ce dernier cas, APDU étendue.
- Support des APDU étendues par certains terminaux. Mais, taille maximale gérée nettement limitée (par exemple 800 octets).

Accès illégal en écriture

Un terminal trop laxiste dans sa gestion mémoire ou la réception des blocs est vulnérable à un débordement de tampon.

Outline

Limitation des outils existants

- **Javacard**. Partiellement programmable. Couche de transport implémentée. Limitées aux attaques via les APDUs étendues.
- **Fun Card**, **Prussian Card**, **BasicCard**. Totalement programmable mais via un lecteur de carte à puce.
- **Emulateur physique de carte à puce**. Port externe pour la programmation. Mais problème de coût, de degré de personnalisation, de temps de prise en main ou de difficulté à l'automatiser.

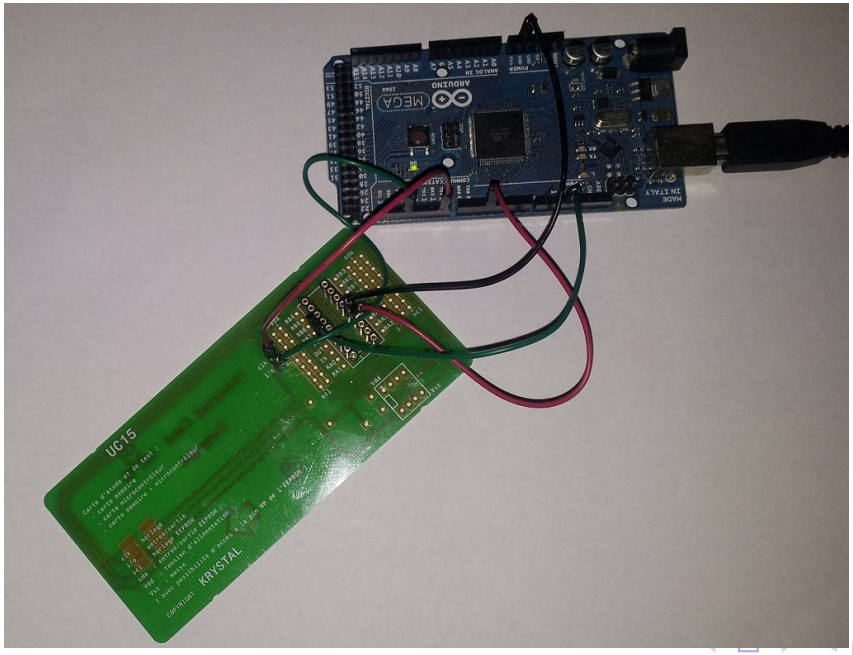
Développement d'un émulateur

Utilisation d'une plateforme Arduino

- Faible coût, facilité de prise en main, communauté active.
- Kit de développement très complet, libre, utilisable sous Eclipse.
- Connexion USB : programmation et pilotage sur un PC via python . . . automatisation de tests de fuzzing !

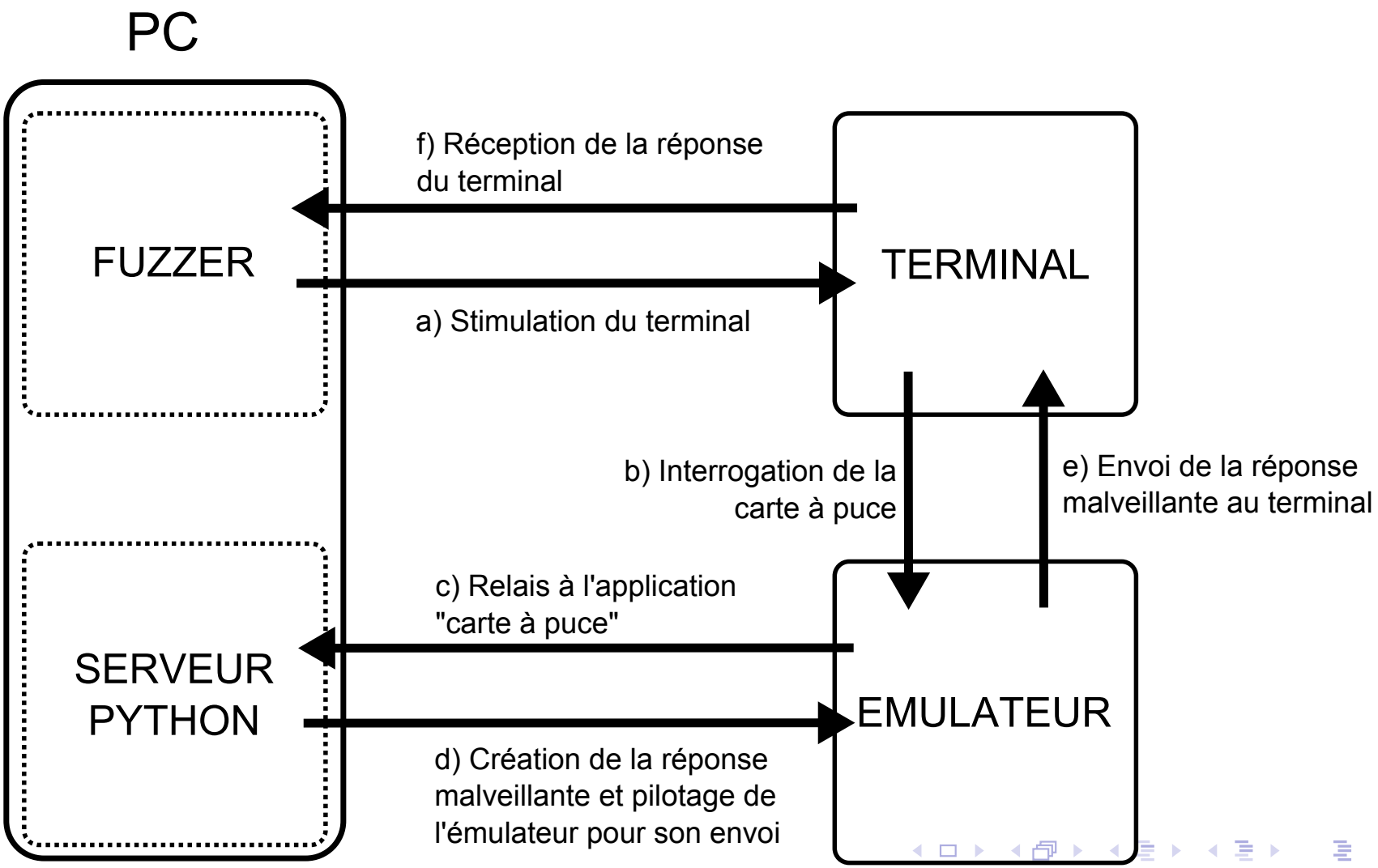
Depuis la réalisation de notre outil, apparition d'alternatives : Pinguino, Raspberry. . .

Description de l'émulateur



Outline

Pilotage des tests par fuzzing



Outline

Architecture des produits attaqués

Intel MCS-51 (8051)

- Trois types de mémoire de taille réduite :
  - CODE ($<$ 100Ko).
  - XRAM ($<$ 2Ko).
  - IRAM ($=$ 256 octets).
- Architecture Harvard.
- Pile située dans le segment IRAM de 256 octets.

- Utilisation de tableaux globaux pour stocker les commandes/réponses en XRAM.
- Option de compilation : création des variables locales des fonctions en XRAM et non pas dans la pile pour éviter de la saturer trop rapidement.

Architecture des produits attaqués

Intel MCS-51 (8051)

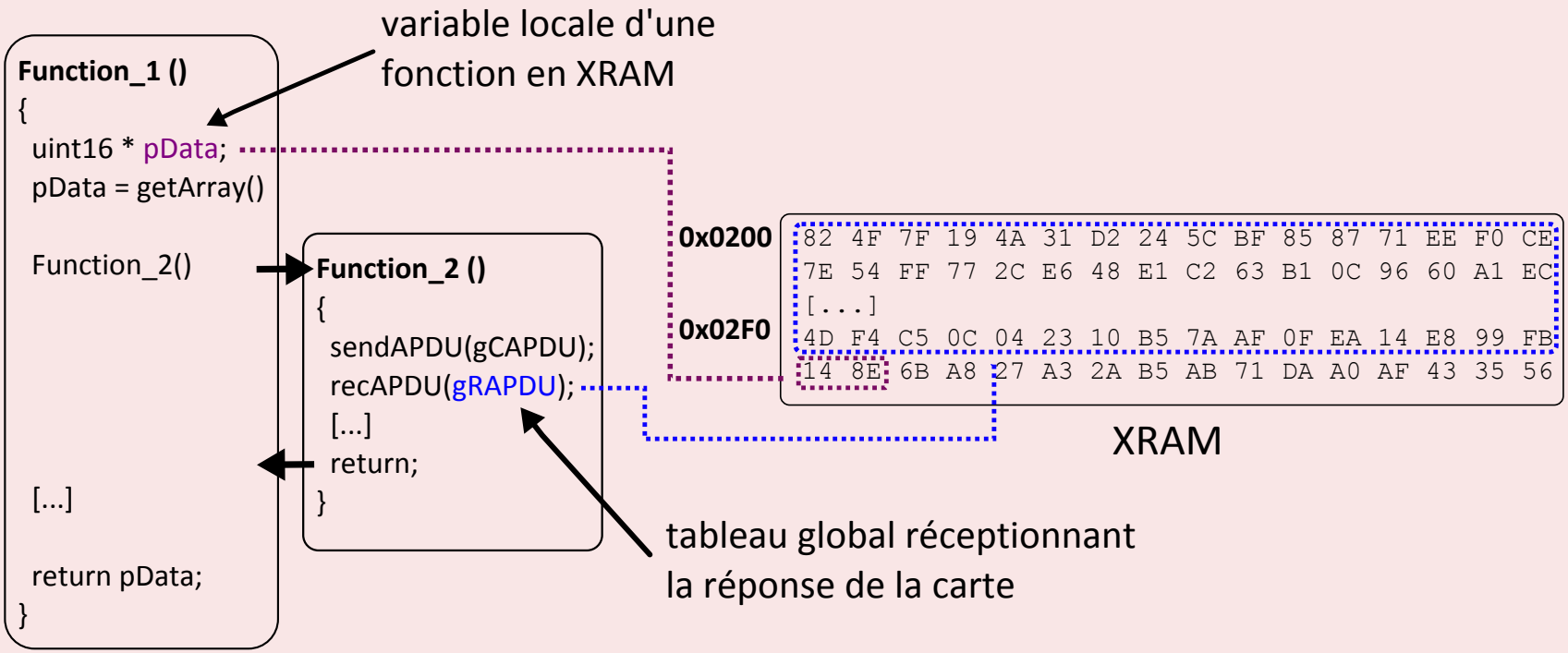
- Trois types de mémoire de taille réduite :
  - CODE ($<$ 100Ko).
  - XRAM ($<$ 2Ko).
  - IRAM ($=$ 256 octets).
- Architecture Harvard.
- Pile située dans le segment IRAM de 256 octets.

- Utilisation de tableaux globaux pour stocker les commandes/réponses en XRAM.
- Option de compilation : création des variables locales des fonctions en XRAM et non pas dans la pile pour éviter de la saturer trop rapidement.

Vecteur d'attaque privilégié

Ecrasement de variable en XRAM

La réponse APDU en XRAM écrase une variable globale ou une variable locale utilisée par une fonction appelante.



Outline

Résultats obtenus

5 terminaux testés.

- Attaque en boîte noire (dans certains cas, retours avec les développeurs).
- Aucun mode de debug disponible.

4 failles découvertes sur trois terminaux :

- 1 basée sur un ATR mal formaté.
- 1 basée sur les octets de procédure du mode T0.
- 2 basées sur des APDUs étendus.

Dump via ATR mal formaté

Récupération d'une partie de la réponse APDU précédente

- **3B 0F** *00 00 00 00 27 01 00 03* 00 01 02 03 04 05 06
- Tableau de réception de l'ATR partagé avec le tableau de réception de la réponse APDU.

Comportement différent lors de la saisie sécurisée d'un code PIN

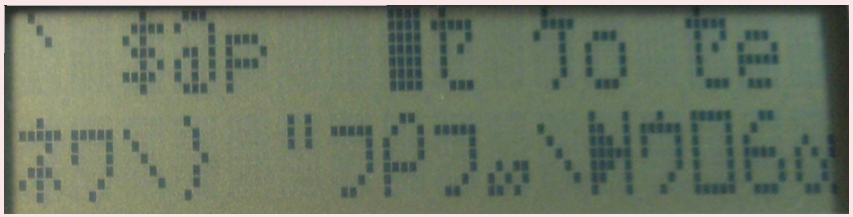
- **3B 0F** *00 00 00 00 27 01 00 03* 00 00 00 00 00 00 00
- Tableau de réception de l'ATR partagé avec la commande Verify PIN du terminal, mais effacement de cette dernière.
- Mécanisme crucial pour la sécurité du produit.

Dump par attaque sur les octets de procédure T0

Corruption de l'écran lors des attaques

Dump par attaque sur les octets de procédure T0

Récupération du code du firmware



Upper 4 Bits / Lower 4 Bits	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
xxxx0000	CG RAM (1)			0	@	P	`	p				ー	タ	ミ	α	p
xxxx0001	(2)		!	1	A	Q	a	q			。	ア	チ	ム	ä	q

- Conversion :
  - 60 XX 24 40 70 XX XX 8E BE XX B9 6F XX BE 65 XX
  - C8 DC 12 7D XX 22 CC EF CC EC 12 1A B3 91 36 E0
- Faille basée sur le firmware du micro-contrôleur.

Dump par attaque via les APDUs étendues

Dump de la totalité de la XRAM

- Ecrasement du pointeur des données CCID à renvoyer.
- Récupération du tableau de réception des commandes APDU.
- Faille basée sur une bibliothèque du firmware du micro-contrôleur . . .qui indiquait pourtant explicitement se protéger contre les débordements de tampon !

Dump par attaque via les APDUs étendues

Commande APDU déclenchant notre charge — Fragment d'une précente commande Verify PIN

0x0000	80	A8	00	00	08	24	12	34	FF	FF	FF	FF	FF	FF	05	BA
	BA	06	BA	07	BA	08	BA	09	BA	0A	BA	0B	BA	0C	BA	0D
	[...]															
	BA	76	BA	77	BA	78	BA	79	BA	7A	BA	7B	BA	7C	BA	7D
	BA	7E	BA	7F	00	33	44	55	00	00	00	11	22	33	44	55

Tableau du terminal réceptionnant les commandes APDU

0x0105	80	02	00	00	00	00	11	00	00	00	00	55	4D	50	54	45
	53	54	02	44	55	4D	50	54	45	53	54	02	44	55	4D	50
	54	45	53	54	03	44	55	4D	50	54	45	7B	53			
	[...]															
	44	55	4D	50	54	45	53	54	26	44	55	4D	50	54	45	00
	00	00	00	00	01	7E	02	00	00	00	00	00	00	00	00	00

Tableau contenant la réponse CCID

0x0350	[...]															
	00	00	00	00	00	00	00	00	00	00	00	00	90	00	9F	53
	54	24	44	55	4D	50	54	45	53	54	25	44	55	4D	50	54
	45	53	54	26	44	55	4D	50	54	45	53	54	27	44	55	4D
	03	00	17	BA	1C	BA	1D	BA	1E	BA	1F	BA	20	BA	21	BA
	22	BA	23	BA	24	BA	25	BA	26	BA	27	BA	28	BA	29	BA

Tableau contenant des fragments de commande et de réponse APDU

Outline

Résultats obtenus

ISO/IEC 7816-3, un vecteur d'attaque à prendre en compte :

- 4 failles découvertes.

Des failles exploitables, même sur architecture 8051 :

- 3 failles exploitées, dont :
  - Dump du code sur l'écran.
  - Dump de la totalité de la RAM permettant de lire la totalité d'une précédente transaction.
- 2 failles indépendantes des développeurs, mais reposant sur une faille des bibliothèques applicatives du micro-contrôleur !

Applications futures

Applicable sur tout produit avec une carte en contact

- bancaire, One Time Password (OTP), et même les téléphones (carte SIM).
- architecture plus complexe (Windows CE, Linux, Android ...) : augmentation de la surface d'attaque

Questions ?