Projekt Ventures

Dokumentacja Techniczna Wypracowanych Rozwiązań

Lucjan Hanzlik

Spis treści

1 Wstęp

Niniejsza dokumentacja przedstawia wysopoziomy opis protokołów wypracowanych w ramach projektu, jak również szczegółowe wytyczne dotyczące ich implementacji tj. opis używanych struktur danych (w notacji ASN.1) oraz komend APDU wymienianych pomiędzy czytnikiem i kartą. W celu pełnego zrozumienia niniejszej dokumentacji należy posiłkować się dokumentacjami [1], [2] i [3].

1.1 Używane skróty

Nazwa	Skrót
Algorytm deszyfrujący	**DEC**
Algorytm szyfrujący	**ENC**
Centrum certyfikacji	CA
Certyfikat karty (dla klucza PK_{PICC})	$cert_{PICC}$
Efemeryczny klucz prywatny	$\widetilde{SK}$
Efemeryczny klucz publiczny	$\widetilde{PK}$
Hasło nadrukowane na powierzchni dokumentu	**CAN**
Klucz prywatny karty	SK_{PICC}
Klucz publiczny karty	PK_{PICC}
Kod uwierzytelniania wiadomośc	**MAC**
Funkcja pairingowa (z ang. pairing function)	**e**
Funkcja mapująca	**MAP**
Funkcja skrótu	**H**
Funkcja tworzenia klucza (Key Derivation Function)	**KDF**
Funkcja uzgadniania klucza (Key Agreement Function)	**KA**
PIN	**PIN**
PUK	**PUK**
Warstwa danych czytelna przez czytnik optyczny	**MRZ**

Uwaga: Do zrozumienia niniejszej dokumentacji wymagana jest podstawowa wiedza na temat kryptografii asymetrycznej oraz kart mikroprocesorowych.

1.2 Wykorzystywane algorytmy matematyczne

W niniejszej dokumentacji wykorzystywane będą podstawowe operacje obliczeniowe w ciałach skończonych oraz dla krzywych eliptycznych. Poniżej zamieszczone zostały pseudokody dla części z nich. Więcej informacji można znaleźć w dokumentacji [1].

1.2.1 Potęgowanie w ciałach skończonych $\mathbb{Z}_p$

Jedną z wielu metod liczenia potęg jest potęgowanie przez kwadratowanie (algorytm szybkiego potęgowania).

Algorytm potęgowanie w $\mathbb{Z}_p$ jest wykorzystywany przez protokół, uzgadniania klucza, Diffiego-Hellmana.

1.2.2 Operacje grupowe dla krzywej eliptycznej $y^2 = x^3 + ax + b \mod p$

Najczęściej wykorzystywanymi operacjami na krzywych eliptycznych, w kryptografii, są dodawanie punktów (odpowiednik mnożenia modulo w $\mathbb{Z}_p$) oraz mnożenie punktu przez skalar (odpowiednik potęgowania modulo w $\mathbb{Z}_p$). Poniższy algorytm opisuje dodawanie punktów. Przyjmijmy, że $\mathcal{O}$ oznacza punkt w nieskończoności dla omawianej krzywej.

Mnożenie punktu przez skalar jest analogiczną (dla grup addytywnych) operacją do potęgowania. Stąd można wykorzystać algorytm podobny do algorytmu 1.

Input : Podstawa g, wykładnik x, moduł p
Output: $g^x \mod p$

```
1  begin
2      w = 1
3      for wszystkich cyfr rozwinięcia dwójkowego liczby x zaczynając od najbardziej znaczącej
       do
4          if cyfra jest zerem then
5              w = w · w mod p
6          end
7          else
8              w = w · w · x mod p
9          end
10     end
11     return w
12
```

Algorithm 1: Algorytm szybkiego potęgowania w $\mathbb{Z}_p$

Input : Punkt $P = (x_P, y_P)$, Punkt $Q = (x_Q, y_Q)$, Parametry krzywej (a, b, p)
Output: Wynikowy punkt $R = (x_R, y_R)$

```
1  begin
2      if P == O then
3          return Q
4      end
5      if Q == O then
6          return P
7      end
8      if x_P == x_Q and y_p == −y_Q then
9          return O
10     end
11     else if P == Q then
12         λ = (3x_P^2 + a)/(2y_P) mod p
13         x_R = λ^2 − 2x_P mod p
14         y_R = λ(x_P − x_R) − y_P mod p
15         return (x_R, y_R)
16     end
17     else if P ≠ Q then
18         λ = (y_Q − y_P)/(x_Q − x_P) mod p
19         x_R = λ^2 − x_P − x_Q mod p
20         y_R = λ(x_P − x_R) − y_P mod p
21         return (x_R, y_R)
22     end
23
```

Algorithm 2: Dodawanie punktów na krzywej $y^2 = x^3 + ax + b \mod p$

```
   Input  : Punkt P = (x_P, y_P), Skalar s, Parametry krzywej (a, b, p)
   Output: Wynikowy punkt R = (x_R, y_R)
1  begin
2      R = O
3      for wszystkich cyfr rozwinięcia dwójkowego liczby s zaczynając od najbardziej znaczącej
       do
4          if cyfra jest zerem then
5              R = R + R
6          end
7          else
8              R = R + P
9          end
10     end
11     return R
12
```

Algorithm 3: Mnożenie punktu przez skalar na krzywej $y^2 = x^3 + ax + b \mod p$

1.3 Funkcja pairingowa

W niniejszej dokumentacji funkcje pairingową definiujemy nad krzywymi eliptycznymi Barreto-Naehrig (BN). Nad krzywymi BN można zdefiniować funkcję pairingową $\mathbf{e} : E(\mathbb{F}_p) \times E'(\mathbb{F}_{p^2}) \to \mathbb{F}_{p^{12}}$, dzięki czemu część operacji można wykonywać na krzywej nad ciałem $\mathbb{F}_p$. Szczegółowy opis obliczania funkcji pairingowej nad krzywymi BN został opisany w [3].

2 Wysokopioziomowy opis protokołów

Protokoły opisane w tym rozdziale pozwalają nawiązać bezpieczną komunikację pomiędzy kartą, a czytnikiem, korzystając jedynie z niedużego hasła (π) znanego przez obie stron protokołu. Wspólnym hasłem może być PIN wprowadzany przez użytkownika lub hasło nadrukowane na powierzchni karty. Dodatkową własnością opisanych protokołów jest autoryzacja karty tzn. karta potwierdza, poprzez znajomość klucza SK_{PICC}, że otrzymała klucz publiczny PK_{PICC} certyfikowany przez CA.

Dla uproszczenia, w poniższych opisach wykorzystywana jest notacja multiplikatywna, a nie notacja addytywna. Protokoły zachowują jednak swoje właściwości również w przypadku zastosowanie krzywych eliptycznych. Jeżeli nie zaznaczono inaczej to wszystkie poniższe operacje są przeprowadzanie w ciele $\mathbb{F}_p$.

2.1 Funkcja pairingowa

Niech $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ będą grupami cyklicznymi rzędu q (zakładamy, że $\mathbb{G}_1$, $\mathbb{G}_2$ to krzywe elityczne, a $\mathbb{G}_T$ to grupa cykliczna w ciele skończonym). Niech funkcja dwuliniowa $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ ma następujące właściwości:

- dla każdego $g \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ i $a, b \in \mathbb{Z}_q$ mamy $\mathbf{e}(g^a, g_2^b) = \mathbf{e}(g, g_2)^{a \cdot b}$,
- jeżeli g generuje $\mathbb{G}_1$ i g_2 generuje $\mathbb{G}_2$ to $\mathbf{e}(g, g_2)$ generuje $\mathbb{G}_T$,
- istnieje efektywny algorytm liczenia $\mathbf{e}(y, y_2)$, dla dowolnych $y \in \mathbb{G}_1$ i $y_2 \in \mathbb{G}_2$.

Taką funkcję $\mathbf{e}$ nazywamy w kryptografii funkcją pairingową.

2.2 Funkcja mapująca

Funkcja ta jest wykorzystywana do interaktywnego tworzenia efemerycznych parametrów na podstawie sekretu s oraz statycznych parametrów D_{PICC}. Istnieje kilka rodzajów funkcji mapującej (zobacz [4]). W niniejszym dokumencie skupimy się na jednej z nich, a mianowicie funkcji Generic Mapping (GM).

Uwaga: W skład parametrów, w przypadku kryptografii ciał skończonych, wchodzą: generator grupy cyklicznej g, rząd grupy q oraz moduł p. W przypadku krzywych eliptycznych to odpowiednio: generator grupy cyklicznej G, rząd grupy q, moduł p oraz współczynniki a i b, gdzie $y^2 = x^3 + ax + b \mod p$ (dla punktów $(x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p$) to wzór wykorzystywanej krzywej. W przypadku protokołu PPACE | AA dodatkowymi parametrami są generator g_2, współczynniki a_2, b_2 opisujące grupę $\mathbb{G}_2$ oraz moduł p_T dla grupy $\mathbb{G}_T$.

Karta (PICC)		Czytnik (PCD)
skopiuj D_{PICC} jako $\widetilde{D}$		skopiuj D_{PICC} jako $\widetilde{D}$
wybierz $y_A \leftarrow \mathbb{Z}_q^*$		wybierz $y_B \leftarrow \mathbb{Z}_q^*$
$Y_A := g^{y_A}$		$Y_B := g^{y_B}$
	$\xleftarrow{Y_B}$	
	$\xrightarrow{Y_A}$	
jeśli $Y_B \notin \langle g \rangle \setminus \{1\}$ przerwij		jeżeli $Y_A \notin \langle g \rangle \setminus \{1\}$ przerwij
$h := Y_B^{y_A}$		$h := Y_A^{y_B}$
$\hat{g} := h \cdot g^s$		$\hat{g} := h \cdot g^s$
użyj $\hat{g}$ jako generatora w $\widetilde{D}$		użyj $\hat{g}$ jako generatora w $\widetilde{D}$
zwróć $\widetilde{D}$		zwróć $\widetilde{D}$

Generic Mapping

2.3 Simplified PACE | AA

Protokół SPACE | AA korzysta z funkcji Generic Mapping w celu uzgodnienia efemerycznych parametrów oraz korzysta z wykorzystywanej w niej losowości. Wykorzystanie tej funkcji zostało zaznaczone na poniższym przebiegu.

Karta (PICC)		Czytnik (PCD)
statyczne parametry D_{PICC}		
wybierz losowo $s \leftarrow Dom(\mathbf{ENC})$		
$K_\pi := \mathbf{KDF}_\pi(\pi)$		$K_\pi := \mathbf{KDF}_\pi(\pi)$
$z := \mathbf{ENC}(K_\pi, s)$		
	$\xrightarrow{D_{PICC}, z}$	przerwij jeżeli D_{PICC} niepoprawne
		$s := \mathbf{DEC}(K_\pi, z)$
	Generic Mapping	
wybierz $y_A \leftarrow \mathbb{Z}_q^*$		wybierz $y_B \leftarrow \mathbb{Z}_q^*$
$Y_A := g^{y_A}$		$Y_B := g^{y_B}$
	$\xleftarrow{Y_B}$	
	$\xrightarrow{Y_A}$	
przerwij jeżeli $Y_B \notin \langle g \rangle \setminus \{1\}$		przerwij jeżeli $Y_A \notin \langle g \rangle \setminus \{1\}$
$h := Y_B^{y_A}$		$h := Y_A^{y_B}$
$\hat{g} := h \cdot g^s$		$\hat{g} := h \cdot g^s$
wybierz $y'_A \leftarrow \mathbb{Z}_q^*$		wybierz $y'_B \leftarrow \mathbb{Z}_q^*$
$Y'_A := \hat{g}^{y'_A}$		$Y'_B := \hat{g}^{y'_B}$
	$\xleftarrow{Y'_B}$	
	$\xrightarrow{Y'_A}$	
sprawdź czy $Y'_B \neq Y_B$		sprawdź czy $Y'_A \neq Y_A$
$K := \mathbf{KA}(y'_A, Y'_B, \widetilde{D})$		$K := \mathbf{KA}(y'_B, Y'_A, \widetilde{D})$
$K_{ENC} := \mathbf{KDF}(K, 1)$		$K_{ENC} := \mathbf{KDF}(K, 1)$
$K_{MAC} := \mathbf{KDF}(K, 2)$		$K_{MAC} := \mathbf{KDF}(K, 2)$
$T_A := \mathbf{MAC}(K_{MAC}, (Y'_B, \mathcal{G}))$		$T_B := \mathbf{MAC}(K_{MAC}, (Y'_A, \mathcal{G}))$
	$\xleftarrow{T_B}$	
	$\xrightarrow{T_A}$	
przerwij jeżeli T_B niepoprawne		przerwij jeżeli T_A niepoprawne
$w := SK_{PICC}/y_A$		
$\sigma := \mathbf{ENC}(K_{ENC}, w \Vert cert_{PICC})$		
	$\xrightarrow{\sigma}$	
		$(w, cert_{PICC}) = \mathbf{DEC}(K_{ENC}, \sigma)$
		sprawdź certyfikat $cert_{PICC}$ i ekstrahuj z niego PK_{PICC}
		jeżeli $PK_{PICC} \neq (Y_A)^w$ przerwij

Protokół SPACE | AA

2.4 Pairing PACE | AA

Protokół Pairing PACE | AA korzysta z dowolnej funkcji mapującej, korzystającej ze wspólnego dla czytnika i karty hasła, parametry statyczne D_{PICC} na parametry efemeryczne $\widetilde{D}$. Z powodu wykorzystania funkcji dwuliniowej **e**, protokół ten można używać jedynie korzystając z kryptografii krzywych eliptycznych.

Karta (PICC)		Czytnik (PCD)
statyczne parametry D_{PICC}		
wybierz losowo $s \leftarrow Dom(\mathbf{ENC})$		
$K_\pi := \mathbf{KDF}_\pi(\pi)$		$K_\pi := \mathbf{KDF}_\pi(\pi)$
$z := \mathbf{ENC}(K_\pi, s)$		
	$\xrightarrow{D_{PICC}, z}$	przerwij jeżeli D_{PICC} niepoprawne
		$s := \mathbf{DEC}(K_\pi, z)$
$\widetilde{D} := \mathbf{MAP}(D_{PICC}, s)$		$\widetilde{D} := \mathbf{MAP}(D_{PICC}, s)$
ekstrahuj $\hat{g}$ z $\widetilde{D}$		ekstrahuj $\hat{g}$ z $\widetilde{D}$
wybierz $y'_A \leftarrow \mathbb{Z}_q^*$		wybierz $y'_B \leftarrow \mathbb{Z}_q^*$
$Y'_A := \hat{g}^{y'_A}$		$Y'_B := \hat{g}^{y'_B}$
	$\xleftarrow{Y'_B}$	
	$\xrightarrow{Y'_A}$	
sprawdź czy $Y'_B \neq Y_B$		sprawdź czy $Y'_A \neq Y_A$
$K := \mathbf{KA}(y'_A, Y'_B, \widetilde{D})$		$K := \mathbf{KA}(y'_B, Y'_A, \widetilde{D})$
$K_{ENC} := \mathbf{KDF}(K, 1)$		$K_{ENC} := \mathbf{KDF}(K, 1)$
$K_{MAC} := \mathbf{KDF}(K, 2)$		$K_{MAC} := \mathbf{KDF}(K, 2)$
$T_A := \mathbf{MAC}(K_{MAC}, (Y'_B, \mathcal{G}))$		$T_B := \mathbf{MAC}(K_{MAC}, (Y'_A, \mathcal{G}))$
	$\xleftarrow{T_B}$	
	$\xrightarrow{T_A}$	
przerwij jeżeli T_B niepoprawne		przerwij jeżeli T_A niepoprawne
$w := SK_{PICC}/y'_A$		
$\sigma := \mathbf{ENC}(K_{ENC}, w \Vert cert_{PICC})$		
	$\xrightarrow{\sigma}$	
		$(w, cert_{PICC}) = \mathbf{DEC}(K_{ENC}, \sigma)$
		sprawdź certyfikat $cert_{PICC}$ i ekstrahuj z niego PK_{PICC}
		jeżeli $\mathbf{e}(\hat{g}, PK_{PICC}) \neq \mathbf{e}((Y'_A)^w, g_2)$ przerwij

Protokół PPACE | AA

3 Wykorzystywane algorytmy kryptograficzne i struktury danych ASN.1

3.1 Pliki na karcie

W celu umożliwienia odpowiedniego doboru algorytmów przez czytnik, karta musi posiadać pliki zawierające zbiór. Plik `EF.CardAccess` (opis pliku w tabeli 4) powinien zawierać zbiór dostępnych protokołów bezpieczeństwa

```
SecurityInfos ::= SET OF SecurityInfo.
```

Zbiór ten powinien zawierać struktury `PACEInfo` oraz `PACEDomainParameterInfo` (szczegółowy opis struktur jest w rozdziale A.1.1.1 dokumentu [2]). Pierwsza struktura pozwala używać zdefiniowanych w tabeli 4 w dokumencie [2] parametrów systemu. Druga z kolei pozwala definiować niestandardowe parametry. Dla protokołu Pairing PACE | AA należy używać jedynie struktury `PACEDomainParameterInfo` oraz kryptografii krzywych eliptycznych.

Uwaga: W momencie pisanie dokumentacji nie wyspecyfikowano odpowiednich Object Identifiers dla protokołów SPACE | AA i PPACE | AA.

Nazwa pliku	EF.CardAccess
ID pliku	0x011C
Skrócone ID	0x1C
Odczyt	zawsze
Zapis	nigdy
Długość	zmienna
Zawatość	Struktura `SecurityInfos` zakodowana w DER

Tablica 4: Opis pliku EF.CardAccess

3.2 Certyfikaty

Certyfikat $cert_{PICC}$ używany przez kartę w obu protokołach, powinien mieć strukturę CVC (Card Verifiable Certificate). Szczegółowy opis CVC można znaleźć w rozdziale C dokumentu [2].

3.3 Algorytmy

3.3.1 Funkcja tworzenia kluczy KDF

Niech $\mathbf{KDF}_\pi(\pi) = \mathbf{KDF}(f(\pi), 3)$, gdzie funkcja kodowania $f()$ jest zdefiniowana w tabeli 5.

Input : Klucz K, licznik c
Output: tablica bajtów
1 **begin**
2 | **return** $\mathbf{H}(K||c)$
3

Algorithm 4: KDF

Rodzaj hasła	Kodowanie
MRZ	SHA-1(Numer seryjny dokumentu ‖ Data urodzin ‖ Data wygaśnięcia dokumentu)
CAN	Ciąg bajtów konwertowanych zgodnie z tabelą 24 w dokumencie [2]
PIN	Ciąg bajtów konwertowanych zgodnie z tabelą 24 w dokumencie [2]
PUK	Ciąg bajtów konwertowanych zgodnie z tabelą 24 w dokumencie [2]

Tablica 5: Kodowanie hasła

3.3.2 Funkcja uzgadniania klucza

Funkcja, która na podstawie załączonych parametrów D, klucza publicznego PK i klucza prywatnego SK wylicza klucz K. Tabela 6 opisuje format kluczy PK, SK i K oraz algorytm wyliczania K.

Algorytm / Format	Kryptografia ciał skończonych $\mathbb{F}_p$	Krzywe eliptyczne
Algorytm	PKCS#3	ECKA
Format klucza publicznego	X9.42 [5]	ECC [1]

Tablica 6: Funkcja uzgadniania klucza

3.3.3 Algorytm szyfrowania

Należy używać jedynie algorytmu szyfrowania AES (128, 192 i 256) w wersji CBC z IV=0. W celu stworzenia klucza dla AES-128 należy w algorytmie **KDF** skorzystać z funkcji skrót SHA-1 i jako klucz wykorzystać jedynie 16 pierwszych bajtów jako klucz dla AES (SHA-1 zwraca 20 bajtów).

W przypadku AES-192 i AES-256 należy skorzystać z funkcji skrótu SHA-256, w przypadku AES-192 należy skorzystać z 24 pierwszych bajtów (SHA-256 zwraca 32 bajty danych).

3.3.4 Kod uwierzytelnienia wiadomości

Należy używać kodów uwierzytelnienia opartych o algorytmy szyfrowania (tzw. CMAC). W tym celu należy używać algorytmu szyfrowania AES w wersji CMAC. Wykorzystane powinny być jedynie pierwsze 8 bajtów wyniku.

3.3.5 Generowanie krzywych eliptycznych dla PPACE | AA

W protokole PPACE | AA należy korzystać z krzywych Barreto-Naehrig. Krzywe te są parametryzowane jako:

$$\begin{aligned} t(x) &= 6x^2+1 \\ n(x) &= 36x^4-36x^3+18x^2-6x+1 \\ p(x) &= 36x^4-36x^3+24x^2-6x+1. \end{aligned}$$

Należy dobrać parametr x tak, aby $n(x)$ i $p(x)$ były pierwsze.

4 Komendy APDU ISO7816

W niniejszym rozdziale przedstawione zostanie mapowanie protokołów SPACE | AA i PPACE | AA na komendy APDU.

4.1 SPACE | AA i PPACE | AA

Następująca sekwencja komend powinna zostać użyta w celu implementacji protokołów SPACE | AA i PPACE | AA:

- MSE Set:AT
- Sekwencja komend General Authenticate (dane wymieniane w poszczególnych krokach zostały opisane w tabeli 7

Krok	Dane wysyłane przez terminal		Dane wysyłane przez kartę	
1	-	Brak	0x80	Zaszyfrowane hasło dla funkcji mapującej
2	0x81	Dane dla funkcji mapującej	0x82	Dane dla funkcji mapującej
3	0x83	Efemeryczny klucz publiczny	0x84	Efemeryczny klucz publiczny
4	0x85	Kod uwierzytelniający czytnika	0x86	Kod uwierzytelniający karty
5	-	Brak	0x89 0x90	Zaszyfrowany dowód znajomości wykładnika Zaszyfrowany certyfikat CV

Tablica 7: Dane użyte w kolejnych komendach General Authenticate

Uwaga: Jeżeli w pliku EF.CardAccess jest wskazanych więcej parametrów, czytnik musi wybrać parametry komendą MSE Set:AT.

4.1.1 Zaszyfrowane hasło dla funkcji mapującej

Deszyfrowane dane powinny być traktowane jako zapis liczby naturalnej w systemie dwójkowym.

4.1.2 Dane dla funkcji mapującej

W przypadku protokołu SPACE | AA dane te zawierają efemeryczne klucze publiczne stron w postaci:

0x04 | | x | | y, gdzie x i y to beznakowy zapis współrzędnych punkty,

dla krzywych eliptycznych i w postaci:

x, gdzie x to beznakowy zapis elementu ciała skończonego.

W przypadku protokołu PPACE | AA dane te zależą od wykorzystywanej funkcji mapującej.

4.1.3 Efemeryczny klucz publiczny

W przypadku krzywych eliptycznych należy użyć nieskompresowaną postać punktu krzywej bez parametrów. W przypadku kryptografii ciał skończonych należy użyć zapis beznakowy elementu ciała bez parametru.

4.1.4 Kod uwierzytelniający

Kody powinny mieć długość 8 bajtów jak opisano w rozdziale 3.3.4.

4.1.5 Zaszyfrowany dowód znajomości wykładnika

W obu przypadkach (kryptografia krzywych eliptycznych, kryptografia ciał skończonych) dowodem znajomości jest element ciała modulo rząd wykorzystywanej grupy. W zależności od wybranych parametrów, dowód ten może nie odpowiadać wielokrotności rozmiaru bloków używanych przez algorytm szyfrowania. Należy wtedy dopełnić zapis szesnastkowy dowodu zerami do wielokrotności rozmiaru bloku. Podczas deszyfracji należy odciąć dopełnione zera.

4.1.6 Zaszyfrowany certyfikat CV

Zapis szesnastkowy certyfikatu należy dopełnić zerami żeby odpowiadał rozmiarowi bloków używanych przez algorytm szyfrowania.

4.2 MSE:Set AT

Komendę należy używać do wyboru i inicjalizacji protokołów SPACE | AA i PPACE | AA.

Komenda		
INS	`0x22`	Zarządzanie bezpiecznym środowiskiem
P1/P2	`0xC1A4`	SPACE | AA i PPACE | AA
Dane	`0x80`	*Referencja używanych algorytmów* Identyfikator obiektu dla używanego protokołu (pomijana jest wartość etykiety ASN.1 `0x06`).
	`0x83`	*Referencja używanego sekretu* `0x01` MRZ `0x02` CAN `0x03` PIN `0x04` PUK
	`0x84`	*Referencja parametrów* Wskazanie na pametry używane przez protokół. Używane tylko jeżeli istnieje więcej niż jedna para parametrów.
Odpowiedź		
Dane	-	Brak
Status	`0x9000`	*Poprawne wykonanie* Protokół został wybrany i zainicjalizowany.
	`0x6A80`	*Niepoprawne dane* Algorytm nie jest wspierany lub błąd w inicjalizacji.

4.3 General Authenticate

Komendę należy używać do przeprowadzenia protokołów SPACE | AA i PPACE | AA. Należy użyć dane opisane w tabeli 7 jako dynamiczne dane autoryzacyjne.

Komenda		
INS	`0x86`	General Authenticate
P1/P2	`0x0000`	
Dane	`0x7C`	*Dynamiczne dane autoryzacyjne* Obiekty danych zgodne z używanym protokołem.
Odpowiedź		
Dane	`0x7C`	*Dynamiczne dane autoryzacyjne* Obiekty danych zgodne z używanym protokołem.
Status	`0x9000`	*Poprawne wykonanie* Krok protokołu został pomyślnie wykonany.
	`0x6300`	*Błąd autoryzacji* Protokół został przerwany błędem.
	`0x6A80`	*Niepoprawne dane* Podane dynamiczne dane autoryzacyjne niepoprawne.
	`inny`	*Błąd systemu operacyjnego* Protokół został przerwany.

Literatura

[1] BSI: Elliptic Curve Cryptography. Technische Richtlinie TR-03111 v2.0 (2012)

[2] BSI: Advanced Security Mechanisms for Machine Readable Travel Documents 2.1. Technische Richtlinie TR-03110-3 (2012)

[3] Devegili, A.J., Scott, M., Dahab, R.: Implementing cryptographic pairings over barreto-naehrig curves. IACR Cryptology ePrint Archive **2007** (2007) 390

[4] Bender, J., Fischlin, M., Kügler, D.: Security analysis of the pace key-agreement protocol. In Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A., eds.: ISC. Volume 5735 of Lecture Notes in Computer Science., Springer (2009) 33–48

[5] American National Standards Institute: Public key cryptography for the financial services industry: Agreement of symmetric keys using discrete logarithm cryptography. ANSI X9.42-2003 (2003)