VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2021 Damián Vertaľ

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

BEZKONTAKTNÍ MIKROPOČÍTAČOVÁ KARTA JAKO SKRÝŠ PRO GEOKEŠING
CONTACTLESS MICROCOMPUTER CARD AS A HIDING PLACE FOR GEOCACHING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Damián Vertaľ**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Karel Burda, CSc.**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Telekomunikační a informační technika**

Ústav telekomunikací

Student: Damián Vertaľ ***ID:*** 223261

Ročník: 2 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Bezkontaktní mikropočítačová karta jako skrýš pro geokešing

POKYNY PRO VYPRACOVÁNÍ:

Podle podrobné specifikace zadané vedoucím práce implementujte kryptografický protokol pro NFC rozhraní mezi smartfonem a mikropočítačovou kartou. Pro smartfon podle této specifikace navrhněte a naprogramujte jednoduchou herní aplikaci. Vytvořený systém otestujte a zhodnoťte.

DOPORUČENÁ LITERATURA:

[1] Burda K.: Bezkontaktní mikropočítačová karta jako skrýš pro geokešing. Elektrorevue, Brno 2020.

[2] Rankl W., Effing W.: Smart card handbook. John Wiley & Sons, N. York 2004.

Termín zadání: 1.2.2021 ***Termín odevzdání:*** 24.5.2021

Vedoucí práce: doc. Ing. Karel Burda, CSc.

prof. Ing. Jiří Mišurec, CSc.
předseda rady studijního programu

ABSTRAKT

Táto diplomová práca sa zaoberá možnosťou využitia bezkontaktných čipových kariet ako elektronickej skrýše v rámci aktivity známou pod názvom Geokešing. V prvej časti sú bližšie vysvetlené teoretické poznatky o kartách, programovanie čipových kariet, vývoj android aplikácií pre komunikáciu s čipovou kartou prostredníctvom NFC rozhrania a využitie eliptických kriviek pri podpisovaní digitálnych správ. V druhej časti je navrhnutá aplikácia pre Java kartu a mobilná aplikácia pre Android smartfón, ktoré sú schopné navzájom komunikovať a využiť kartu ako skrýšu pre geokešing.

KĽÚČOVÉ SLOVÁ

Geokešing, Java Card, karta, NFC, Android, eliptická krivka, ECDSA

ABSTRACT

This master's thesis focuses on the possibility of using contactless smart cards as an electronic hiding place in an activity known as Geocaching. The first part explains the theoretical knowledge about cards, smart card programming, the development of android applications for communication with the smart card using the NFC interface and usage of eliptic curves to sign digital messages. The second part is dedicated to the design of a Java card application and an Android application, which are able to communicate with each other and use contactless card as hiding place for geocaching.

KEYWORDS

Geocaching, Java Card, card, NFC, Android, eliptic curve, ECDSA

VERTAĽ, Damián. *Bezkontaktní mikropočítačová karta jako skrýš pro geokešing*. Brno, 2021, 72 s. Diplomová práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: doc. Ing. Karel Burda, CSc.

VYHLÁSENIE

Vyhlasujem, že svoju diplomovú prácu na tému „Bezkontaktní mikropočítačová karta jako skrýš pro geokešing“ som vypracoval samostatne pod vedením vedúceho diplomovej práce, s využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej diplomovej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto diplomovej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákonníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam výpisov

Úvod

Geokešing je pomerne nový druh športovej a zároveň zábavnej aktivity. Princíp spočíva v tom, že hráč sa snaží nájsť skrýšu s odmenou na základe GPS súradníc danej skrýše. Pre niekoho to teda môže ponúkať zábavnejšiu formu turistiky. Rozvoj tejto aktivity súvisí hlavne s rozvojom GPS systémov. V neposlednom rade ale samozrejme aj s rozvojom smartfónov. Práve rozvoj smartfónov môžeme považovať za zásadný prelom v tejto aktivite. Hráčovi teda k tejto hre postačuje iba zariadenie, ktoré bežne nosí stále vo svojom vrecku.

U dnešných smartfónov je už úplne bežné, že podporujú rozhranie NFC. Tu sa teda núka možnosť využiť to pri tejto hre. Odmena, ktorú skrýša obsahuje, by mohla byť v elektronickej forme a pomocou smartfónu prostredníctvom NFC by hráč túto odmenu získal. Táto odmena v elektronickej forme by mohla byť uchovávaná napríklad v bezkontaktnej karte.[1]

Táto práca sa zaoberá práve možnosťou využitia bezkontaktnej mikropočítačovej karty ako skrýše pre geokešing. Pre úspešnú realizáciu je potrebné navrhnúť aplikáciu pre bezkontaktnú kartu, ktorá slúži ako skrýš. Taktiež je potrebné navrhnúť mobilnú (Android) aplikáciu, ktorá dokáže pomocou rozhrania NFC komunikovať s touto bezkontaktnou kartou. Komunikácia bude kryptograficky zabezpečená a bude prebiehať podľa kryptografického protokolu bližšie upresneného v špecifikácii práce.

1 Teoretická časť

1.1 Geokešing

Geokešing je pomerne nový druh športovej a zároveň zábavnej aktivity. Princíp spočíva v tom, že hráč sa snaží nájsť skrýšu s odmenou na základe GPS súradníc danej skrýše. Pre niekoho to teda môže ponúkať zábavnejšiu formu turistiky.

1.1.1 História

Vývoj geokešingu priamo súvisí s vývojom systému GPS. Až do roku 2000 bola do systémov GPS pri určovaní polohy umelo vnášaná odchýlka známa aj pod názvom výberový prístup (Selective Availability). V roku 2000 bolo ale oznámené vypnutie tejto odchýlky, čo v praxi samozrejme znamenalo lepšiu presnosť určovania polohy GPS systémom pre civilné obyvateľstvo. To položilo aj základy na rozvoj geokešingu. Dovtedy sa presnosť GPS pohybovala v desiatkách, prípadne až stovkách metrov. Pri takýchto odchylkách by bolo veľmi obtiažné hľadať skrýše s ukrytými predmetmi na základe GPS súradníc. No touto zmenou sa umožnilo fungovanie takejto aktivity.[2]

S námetom na takúto aktivitu prišiel Dave Ulmer, ktorý hneď po zrušení vnášanej odchýlky publikoval poznámku o hre s názvom Stash Game. A hneď deň na to, 3.5.2000, Dave Ulmer umiestnil oficiálne prvú skrýšu v rámci hry s názvom GPS Stash Hunt. Súradnice tejto skrýše boli N 45 17.460 W122 24.800 neďaleko mesta Portland v USA. Skrýša bola nájdená hneď na ďalší deň Bobom Perschauom, ktorý hru nazval Great GPS Stash Hunt. Trvalo len 8 dní od umiestnenia prvej skrýše, kým bola 12.5.2000 skrýša umiestnená aj mimo USA, konkrétne na Novom Zélande. 30.5.2000 Matt Stum navrhol, aby táto aktivita dostala názov Geocaching, čo Dave Ulmer akceptoval a deň na to oficiálne oznámil tento názov, ktorý poznáme dodnes. K dátumu 29.1.2020 bolo na celom svete aktívnych 3 001 062 skrýši.[2]

1.1.2 Fungovanie

Princíp hry je jednoduchý. Pokiaľ chceme začať hľadať skrýše, v prvom rade je nutné registrovať sa na oficiálnej stránke geokešingu (www.geocaching.com). Následne si na mape vieme nájsť skrýše, ktoré by sme chceli nájsť. Po tom, čo si hráč vyberie skrýšu a taktiež ju nájde, je potrebné zaznamenať svoj nález v logbooku tým. Hráč ale nemusí len hľadať založené skrýše, ale môže taktiež založiť vlastnú skrýš.[3]

1.1.3 Skrýša

Skrýše sú definované z rôznych hľadísk. Podľa veľkosti sa skrýše delia do 5 skupín:

- mikro - do 100 ml - využívaná najmä v mestách,
- malá - od 100 ml do 1 l - široký výber pre možnú lokáciu, no zároveň ešte obmedzená kapacita pre predmety,
- stredná - od 1 l do 20 l - ideálna veľkosť, ktorá poskytuje dostatočnú kapacitu a zároveň ešte relatívne dobrá možnosť ukrytia,
- veľká - viac ako 20 l - kapacita pre veľké predmety, no zároveň ťažko ukrývateľná,
- iná - obvykle používaná vtedy, keď je skrýša neobvyklého tvaru - do tejto kategórie spadá aj čipová karta, ktorá by bola použitá ako skrýša.[4]

Skrýše sa ale delia aj podľa ich typu. Ide napríklad o:

- tradičná - najbežnejší variant, skrýša sa nachádza na zadaných súradniciach,
- multi - nie sú zadané súradnice so skrýšou, ale súradnice zastávok, kde sú, indície potrebné k zisteniu súradníc skrýše, alebo ďalšej zastávky pred samotnou skrýšou,
- hádanka / puzzle - pre získanie súradníc je potrebné vyriešiť hlavolam alebo šifru zverejnenú na internete, prípadne priamo v teréne,
- stretávka - súradnice určujú miesto, kde sa v presne definovanom čase uskutoční stretnutie ľudí, ktorí sa zaoberajú geokešingom,
- stretávka s vyčistením prírody od odpadkov - funguje na princípe stretávky, ibaže stretnutie usporiadané z dôvodu čistenia daného miesta od odpadkov,
- Earthcache - na mieste sa nenachádza klasická fyzická skrýša, no po príchode na miesto má hráč získať odpovede na otázky viazané na dané miesto, ktoré sú zverejnené v popise skrýše,
- lab skrýša - slúžia na testovanie nových typov skrýš.[4]

Okrem toho existuje aj niekoľko typov skrýš, ktoré sú síce ešte aktívne, no nedajú sa už zakladať nové. Sú to napríklad:

- webkamera - na mieste sa nenachádza klasická fyzická skrýša, ale na danom mieste je potrebné vytvoriť záber kamerou, ktorá dané miesto monitoruje,
- reverzná skrýša - objekt hľadania môže byť budova, prírodný úkaz alebo napríklad aj technické dielo.[4]

Okrem toho je skrýša definovaná aj svojou obtiažnosťou. Tá definuje, ako náročné je skrýšu nájsť resp. náročnosť rozlúštenia súradníc skrýše. Obtiažnosť sa určuje na základe päťbodovej stupnice:

- 1 - veľmi ľahko identifikovateľné - napríklad pri koreňoch samostatne stojacého stromu,
- 2 - ľahko identifikovateľné - napríklad pri stromoch mimo cestičky,
- 3 - zložitejšie - zvyčajne si vyžadujú kvalitnejšiu prípravu - hlavne pre skrýše

typu hádanka,

- 4 - obtiažné - vyžadujú si vyriešenie zložitejších šifier, prípadne potrebné nájsť veľmi ťažko objaviteľné skrýše,
- 5 - najťažšie - zvyčajne pre skrýše typu hádanka, riešenie šifier je časovo náročné.[4]

Skrýše sú ďalej definované aj náročnosťou terénu, v ktorom sú umiestnené. Určuje sa taktiež na základe päťbodovej stupnice:

- 1 - cesty, chodníky, spevnené cesty, značené turistické cesty, zvyčajne maximálne 150 m od miesta, kde je možné zaparkovať auto,
- 2 - nespevnené cesty, vyšliapané cestičky,
- 3 - terén bez ciest, typicky v strmších kopcoch alebo v miestach zo zhustenou vegetáciou,
- 4 - terén prístupný pre športovo zdatnejších ľudí, veľmi strmé kopce so skalami, prípadne bažinatý terén,
- 5 - veľmi náročný terén vyžadujúci napríklad horolezecké vybavenie.[4]

1.2 Vývoj a typy kariet

Čipové karty sú len jedným z druhov identifikačných kariet, ktoré sú špecifikované normou ISO/IEC 7810. Ešte pred tým, ako sa začali používať čipové karty, tu boli karty embosované a karty s magnetickou páskou.[5]

Princíp embosovaných kariet spočíva v tom, že na karte sú údaje vyrazené v podobe reliéfnych znakov. Dáta sú teda uchovávané iba vo fyzickej forme. Uchovávanie dát v elektronickej forme priniesli až karty s magnetickou páskou. [5]

Čipové karty sú najmladšou formou kariet špecifikovaných v ISO/IEC 7810. Tieto karty už obsahujú integrovaný obvod, ktorý je schopný uchovávať, alebo aj spracovávať údaje. Oproti kartám s magnetickými páskami prinášajú značne zvýšenú kapacitu pre uchovávané údaje.[5]

Norma ISO/IEC 7810 špecifikuje 4 rôzne rozmery kariet. Karty rozoberané v tejto práci spadajú do normy ID-1. Rozmerý sú:

- šírka: 85,72 mm,
- výška: 54,03 mm,
- hrúbka: 0,76 mm.[5]

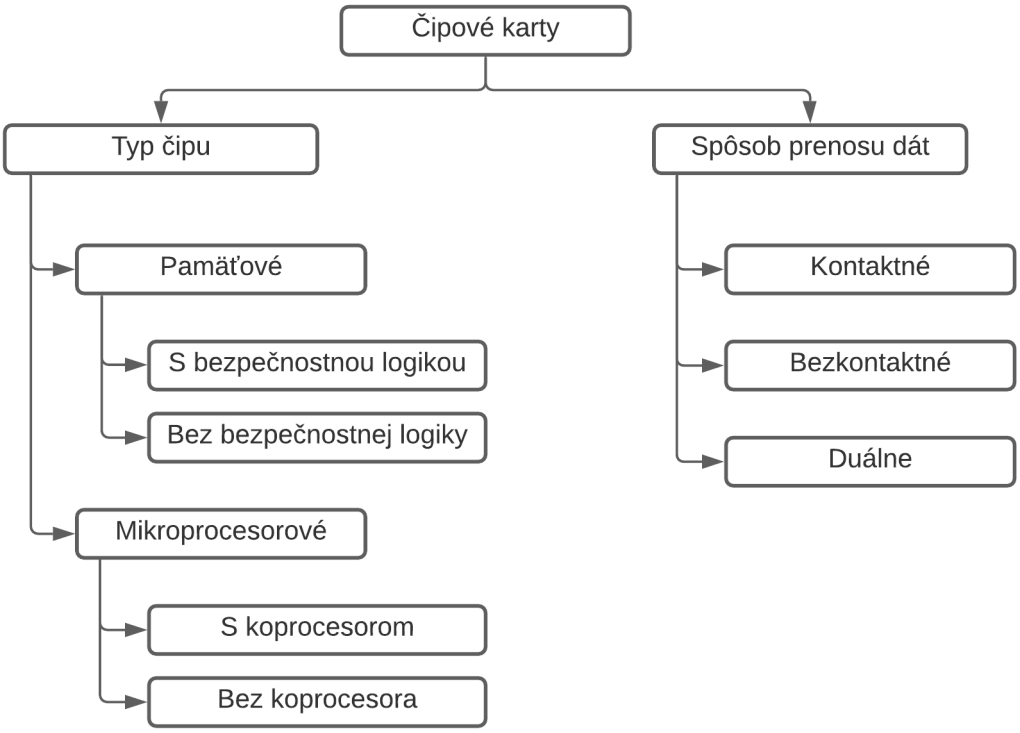
Existujú 2 základné druhy čipových kariet:

- pamäťové čipové karty,
- mikroprocesorové čipové karty.[5]

Nakoľko si praktická časť tejto práce vyžaduje, aby čipová karta obsahovala vlastnú aplikáciu pre komunikáciu s terminálom (smartfónom), použitá bude mikroprocesorová čipová karta.

1.2.1 Čipové karty

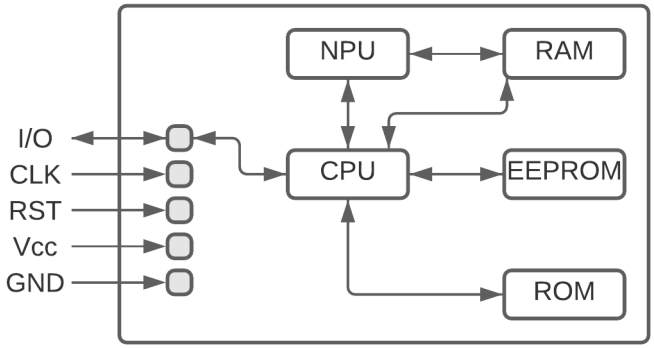
Existuje viacero typov čipových kariet. Rozdeliť sa dajú na základe viacerých kritérií. Na schéme na obr. 1.1 je možné vidieť rozdelenie na základe typu čipu a na základe spôsobu prenosu dát.



Obr. 1.1: Rozdelenie typov čipových kariet na základe typov čipu a spôsobe prenosu dát.

Pamäťové karty sú zvyčajne optimalizované pre nejakú konkrétnu aplikáciu s fixnými operáciami. Údaje dokážu iba uchovávať, nevedia s nimi vykonávať žiadne operácie. Obsahujú pamäť EEPROM. Tam sa dáta ukladajú. Tieto dáta môžu, ale nemusia byť chránené. Pokiaľ sú nejakým spôsobom chránené, ide o pamäťové karty obsahujúce bezpečnostnú logiku. Zväčša bezpečnostná logika predstavuje iba ochranu proti zápisu resp. vymazaniu údajov pamäte, alebo niektorých oblastí pamäte. Existujú však aj pamäťové karty, ktoré dokážu vykonávať aj jednoduché šifrovanie. Najbežnejším príkladom pamäťových kariet sú napríklad skipasy, klubové karty a pod.[5]

Ako už vyplíva z názvu, mikroprocesorové karty obsahujú mikroprocesor, ktorý karte umožňuje dáta už nejakým spôsobom spracovávať. Architektúra mikroprocesorovej karty je znázornená na na obr. 1.2.[5]



Obr. 1.2: Architektúra mikroprocesorových kontaktných kariet.

Takéto karty obsahujú aj operačný systém. Ten je uchovaný v ROM. V EEPROM sú zase uchovávané uložené dáta, ale taktiež aj programový kód aplikácie. RAM je pracovná pamäť procesora, z ktorej sa tak isto, ako pri iných zariadeniach, dáta pri vypnutí napájania stratia.[5]

Mikroprocesorové karty môžeme označiť aj ako programovateľné karty. Programový kód aplikácie nemôže obsahovať vlákna. To teda znamená, že v jednom momente sa dá na danej karte používať iba jedna aplikácia. Avšak je možné do karty implementovať viac rôznych aplikácií.[5]

Mikroprocesorové karty, ktoré obsahujú koprocesor môžeme označovať aj ako kryptografické karty. Tento koprocesor umožňuje karte pre bezpečný prenos dát používať rôzne kryptografické algoritmy, ako napríklad AES, RSA, ECDSA alebo ECDH.[5]

Pokiaľ ide o karty kontaktného typu, ich spojenie s terminálom sa realizuje prostredníctvom 8 kontaktov. Tieto kontakty su na takýchto kartách viditeľné (viď. obr. 1.3). Na obr. 1.3 sú taktiež popísané označenia jednotlivých kontaktov a v tab. 1.1 aj ich názvy a funkcie podľa normy ISO 7816-2.[6]

Problémom pri týchto kontaktoch je ich poruchovosť. Kontakty sa môžu ľahko opotrebovať, zaniesť rôznymi nečistotami a podobne. Taktiež existuje riziko poškodenia až zničenia čipu elektrostatickým výbojom. Pri bezkontaktných kartách sa tieto problémy dokážu eliminovať.[5]

Bezkontaktné karty prinášajú však aj množstvo iných výhod. Vzhľadom na to, že pri týchto kartách nie je nutné ich vkladať do čítačky, je ich používanie značne praktickejšie ako ich kontaktná verzia. Rádiové rozhranie, na základe ktorého karta s terminálom komunikuje, pracuje na diaľku niekoľkých centimetrov (bežne cca 10 cm, v prípade použitia výkonnejšieho terminálu až do vzdialenosť 50 - 100 cm).

Obr. 1.3: Jednotlivé kontakty na čipe.

Tab. 1.1: Názvy jednotlivých kontaktov na čipe a ich funkcia

Kontakt	Názov	Funkcia
C1	Vcc	napájacie napätie
C2	RST	vynulovanie vstupu
C3	CLK	hodinový vstup
C4	AUX1	doplnkový kontakt 1
C5	GND	uzemnenie
C6	Vpp	programovacie napätie (už nepoužívané)
C7	I/O	vstup a výstup pre sériovú komunikáciu
C8	AUX2	doplnkový kontakt 2

Z tohto hľadiska teda môžeme terminály pre bezkontaktné karty deliť na základe vzdialenosti potrebnej pre nadviazanie kontaktu.

Pri tých, ktoré pracujú na najkratšie vzdialenosti, môže byť taktiež potrebné vkladanie karty do terminálu. Avšak výhoda oproti kontaktným kartám spočíva v už spomínanej eliminácií technických problémov s kontaktami. Ďalšou výhodou je aj vkladanie karty. Pri kontaktných kartách sa musia kontakty dotýkať presne určených miest čítačky, pri bezkontaktných kartách a termináloch nezáleži od orientácie vloženej karty, čo v neposlednom rade zvyšuje aj používateľský komfort.[5]

Asi najpoužívanejším variantom sú ale terminály, pri ktorých sa karta nemusí vkladať do slotu, ale iba sa k terminálu priloží.[5] Typickým príkladom je dnes už rozšírená bezkontaktná platba kartou u obchodníkov. Tento spôsob použitia bezkontaktnej karty je aj predmetom tejto práce, kedy takýto terminál predstavuje smartfón resp. jeho NFC rozhranie.

Rozšírené sú ale taktiež terminály, ktorým stačí na nadviazanie spojenia väčšia vzdialenosť. V praxi to pre užívateľa znamená, že svoju kartu ani nemusí vyťahovať

z vrecka a prikladať ju k terminálu, stačí ak s ňou prejde turniketom. Typická oblasť využívania je napríklad hromadná doprava.[5]

Výhodou bezkontaktnej karty je aj to, že jej všetky technické komponenty sú ukryté vo vnútri karty a na povrchu vôbec nie sú viditeľné. To v konečnom dôsledku z marketingového hľadiska nijako neobmedzuje dizajn použitý na karte.[5]

Karta, ktorá dokáže využiť aj bezkontaktný aj kontaktný spojovací prvok sa nazýva duálna karta. Jej výhodou je, že sa dá použiť na oboch typoch terminálu v závislosti od konkrétnej situácie a potreby.[5]

1.3 Komunikácia s čipovou kartou

Komunikáciu medzi kartou a terminálom na transportnej úrovni popisuje norma ISO/IEC 7816-3. Definuje transportné protokoly T=0 a T=1.[7]

Protokol aplikačnej vrstvy pre čipové karty je definovaný v ISO/IEC 7816-4. Tento protokol aplikačnej vrstvy je mapovaný do transportného protokolu (napríklad do spomínaných T=0 alebo T=1).[7]

1.3.1 Transportné protokoly

Spomedzi transportných protokolov T=0 a T=1 je rozšírenejší práve starší z tejto dvojice, protokol T=0, ktorý je bajtovo orientovaný. Spôsobené je to napríklad tým, že našiel využitie v GSM kartách. Vyžaduje si taktiež nízke nároky na pamäť. V súčasnosti sa ale v porovnaní s protokolom T=1 považuje za zastaralejší.

Transportný protokol T=1

Protokol T=1 je blokovo orientovaný protokol, takže dáta sú spracovávané po blokoch. Najmenšia dátová jednotka, ktorú je možné prenášať medzi terminálom a kartou je teda jeden blok.[5]

Pre tento protokol je narozdiel od T=0 typické striktné oddelenie vrstiev RM OSI. Možno ho priradiť k transportnej vrstve. Striktné oddelenie vrstiev znamená, že dáta aplikačnej vrstvy je možné úplne transparentne spracovať transportnou vrstvou. To je dôležité najmä pri šifrovaní údajov.[5]

Komunikácia začína tým, že karta najprv pošle terminálu ATR (viac o ATR v časti 1.3.2) alebo po vykonaní úspešného PPS (viac o PPS v časti 1.3.3). Prvý blok posiela terminál a ďalší karta.[5]

T=1 sa však nevyužíva iba na komunikáciu medzi terminálom a kartou, ale taktiež na komunikáciu medzi terminálom a zariadením, ku ktorému je terminál pripojený.[5]

Existuje viacero typov blokov:

- blok I - informačný blok - používa sa na prenos dát aplikačnej vrstvy,
- blok R - blok potvrdenia prijatia - používa sa na potvrdenie úspešného alebo neúspešného prijatia dát,
- blok S - systémový blok - používa sa na riadenie informácií týkajúcich sa samotného protokolu.[5]

Blok, ktorý sa protokolom prenáša je rozdelený na 3 časti. Pole prológu je povinné, taktiež aj pole epilógu. Informačné pole povinné nie je, obsahuje dáta aplikačnej vrstvy. Tieto polia aj s ich ďalším delením sú znázornené na obr. 1.4.[5]

Pole prológu			Informačné pole	Pole epilógu
NAD	PCB	LEN	APDU	EDC
1 B	1 B	1 B	0 - 254 B	0 - 2 B

Obr. 1.4: Štruktúra bloku podľa protokolu T=1.

Ako je vidieť, pole prológu sa delí na 3 časti. Jeho veľkosť je 3 bajty a obsahuje riadiace údaje. NAD je v poradí prvý bajt. Udáva cieľovú a zdrojovú adresu bloku (3 bity pre zdrojovú a 3 bity pre cieľovú). 1 bajt určený pre PCB slúži na dohľad a kontrolu nad prenosovým protokolom. Určuje hlavne typ bloku, ale môže taktiež niesť aj iné doplňujúce informácie. Pole LEN nesie informáciu o veľkosti informačného poľa. [5]

Využitie informačného poľa závisí od typu bloku. Pokiaľ ide o blok I, potom informačné pole slúži ako kontajner pre dáta aplikačnej vrstvy. Pri prenášaní bloku S obsahuje údaje transportného protokolu. V tomto prípade ide o ojedinelý prípad, kedy informačné pole využíva transportná vrstva.[5]

Pokiaľ ide o veľkosť informačného poľa, jeho maximum je 254 bajtov. Terminál aj samotná karta zvyknú mať predvolenú určitú dĺžku informačného poľa, ktorú využívajú. Túto hodnotu je však možné upraviť prostredníctvom špeciálneho bloku S. V praxi sa využívajú dĺžky informačného poľa 50 - 254 bajtov.[5]

Pole epilógu resp. pole EDC je využívané na detekciu chýb. Hodnota, ktorá je v tomto poli prenášaná je vypočítaná na základe ostatných bajtov v bloku. Na detekciu sa pritom môže využívať LRC aj CRC. Informáciu o tom, čo z toho je použité, obsahuje ATR. Pokiaľ by to uvedené nebolo, implicitne sa používa LRC metóda. Treba poznamenať, že pole epilógu neslúži na opravu chýb, ale iba na ich detekciu.[5]

V protokole T=1 je definovaných niekoľko typov čakacích dôb. Tie určujú zariadeniam minimálne a maximálne čakacie intervaly pre rôzne typy akcií medzi terminálom a kartou. To je dôležité hlavne kvôli predchádzaniu blokovania komunikácie pri chybách v prenose. Jednotlivé typy a ich význam je vidieť v tab. 1.2.[5]

Tab. 1.2: Čakacie doby podľa protokolu T=1

Typ čakacej doby	Funkcia
CWT	maximálny interval medzi 2 po sebe idúcimi znakmi v bloku
BWT	maximálny interval medzi posledným bajtom bloku prijatého na kartu a prvým bajtom bloku vráteného kartou
BGT	minimálny interval medzi posledným bajtom bloku prijatého na kartu a prvým bajtom bloku vráteného kartou

Vďaka skutočnosti, že CWT definuje maximálny interval medzi znakmi bloku, sa dá CWT použiť aj k detekovaniu konca bloku. V reálnej prevádzke to však nie je veľmi výhodné kvôli tomu, že to v konečnom dôsledku znižuje rýchlosť prenosu. Preto sa na to radšej využíva počítanie prijatých bajtov, čo sa porovnáva s hodnotou v poli LEN. Ak by však došlo k chybe prenosu tohto poľa, CWT poskytuje vhodnú alternatívu a zabráni tak blokovaniu prenosu.[5]

BWT slúži hlavne k ukončeniu komunikácie medzi terminálom a kartou, pokiaľ karta nereaguje na požiadavku terminálu. Pri BGT ide v podstate o opak BWT. Pokiaľ by karta potrebovala na svoju odpoveď z nejakého dôvodu viac času, ako je určené v BWT, musí poslať terminálu blok S, ktorým žiada o úpravu tohto času. Terminál následne taktiež blokom S potvrdí túto úpravu. V takomto prípade informačné pole obsahuje hodnotu, ktorou sa vynásobí pôvodné BWT a výsledok je upravené BWT. Táto úprava je ale iba jednorázová.[5]

Pri komunikácii medzi kartou a terminálom je niekedy nutné posielať informačné bloky, ktorých dĺžka je väčšia, ako je veľkosť vysielacej / prijímacej medzipamäte. Vtedy sa využíva funkcia nazývaná reťazenie blokov. Princíp spočíva v segmentácií dát aplikačnej vrstvy do blokov s maximálnou možnou veľkosťou. Pole PCB obsahuje bit s názvom M bit, ktorý nesie informáciu o tom, či sa očakáva ešte prenos ďalších segmentov. Pri posielaní všetkých blokov, okrem bloku s posledným segmentom, teda M bit nesie informáciu, že sa očakávajú ďalšie segmenty (M bit má hodnotu 1). Po prijatí bloku obsahujúceho M bit s hodnotou 0 má príjemca informáciu o prijatí všetkych segmentov reťaze dát aplikačnej vrstvy.[5]

Pri detekovaní chyby v prenose existujú 3 fázy synchronizácie, ktorými sa terminál pokúša obnoviť prenos medzi ním a kartou. Pri prvej pôvodný prijímateľ vyžiada

od pôvodného odosielateľa prostredníctvom R bloku opätovné odoslanie chybného bloku. Pokiaľ z nejakého dôvodu nie je týmto spôsobom možné obnoviť komunikáciu, pokračuje sa ďalšou fázou synchronizácie. Tá spočíva v zaslaní požiadavky na resynchronizáciu terminálom za pomoci S bloku. Tak dôjde k vynulovaniu čítačov karty aj terminálu, čím sa v podstate dostanú do stavu, ktorý bežne nasleduje po prenose ATR. Tieto prvé 2 fázy ovplivňujú iba transportnú vrstvu. V prípade zlyhania spomenutých 2 fáz synchronizácie, poslednou (krajnou) fázou je resetovanie čipovej karty. To je už však sprevádzané zásahom aj do aplikačnej vrstvy a komunikácia musí byť obnovená kompletne. Ak by zlyhala aj táto posledná fáza, dôjde k deaktivácií karty a oboznámeniu používateľa so skutočnosťou, že karta je pravdepodobne chybná.[5]

1.3.2 ATR

ATR je je typ správy, ktorú posiela smart karta terminálu bezprostredne po jej resetovaní. Obsahuje viacero informácií o karte, ale aj parametre prenosového protokolu. Podľa ISO/IEC 7816-3 môže mať ATR maximálne 33 bajtov. V praxi je ale kladený dôraz na to, aby veľkosť ATR nedosahovala túto maximálnu dĺžku, ale aby bola veľkosť čo najmenšia možná. Dôvod na to je ten, aby bola karta použiteľná v čo najkratšom možnom čase, čo sa samozrejme dá dosiahnúť odosielaním čo najkratšej správy ATR.[5],[7]

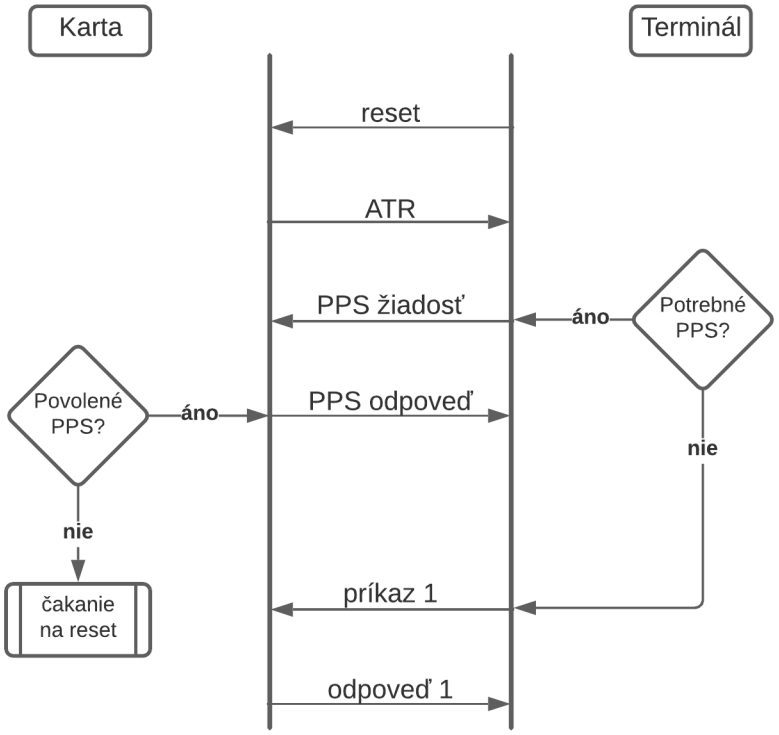
1.3.3 PPS

Ako už bolo spomínané po vykonaní úspšeného PPS začína komunikácia medzi kartou a terminálom. Vykonanie PPS nie je nutnou podmienkou pre komunikáciu, k vykonaniu dochádza iba po vyžiadaní úpravy parametrov prenosu terminálom.[5]

K prípadnému vyžiadaniu PPS terminálom dochádza po úspešnom prijatí ATR terminálom. Pokiaľ si terminál z nejakého dôvodu vyžaduje úpravu niektorého z parametrov, ktoré boli definované v ATR, pošle karte PPS žiadosť. Karta túto úpravu prenosových parametrov môže podporovať, ale nemusí. K úprave teda dochádza iba za predpokladu, že karta umožňuje tieto parametre na základe PPS meniť. Ak túto možnosť podporuje, odosiela terminálu odpoveď. Pokiaľ by túto možnost nepodporovala, v tom prípade karta čaká na reset. Tento proces je znázornený aj na obr. 1.5.[5]

PPS sa delí na niekoľko bajtov. Prvý bajt je označovaný ako PPSS alebo počiatočný znak, vždy nadobúda hodnotu "FF". Pri použití PPS je to povinný bajt, iniciuje PPS.[5]

Za ním nasleduje taktiež povinný znak formátu, alebo PPS0. Ďalšie 3 bajty, ktoré však už nie sú povinné, sa nazývajú znaky parametra, alebo PPS1, PPS2,



Obr. 1.5: Príklad použitia PPS.

PPS3. Tieto bajty kódujú parametre prenosového protokolu, podľa ktorých chce PPS upraviť komunikáciu.[5]

Za nimi nasleduje už iba riadiaci znak PCK, ktorý je teda končený bajt. Ten je opäť povinný. Predstavuje kontrolný súčet dosiahnutý pomocou operácie XOR. Štruktúra je znázornená aj na obr. 1.6.[5]

Obr. 1.6: Štruktúra PPS.

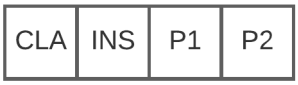
1.3.4 APDU

APDU je dátová jednotka aplikačnej vrstvy pri prenose údajov medzi kartou a terminálom. Je medzinárodné štandardizovaná normou ISO/IEC 7816-4. Existujú 2 typy APDU:

- C-APDU - príkazy posielané na kartu,
- R-APDU - odpovede posielané z karty ako reakcia na C-APDU.[5]

Štruktúra C-APDU

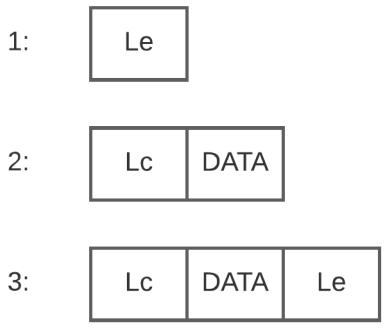
C-APDU sa skladá z hlavičky a tela. Zatiaľ čo hlavička je povinná a má pevne danú dĺžku, telo má premenlivú dĺžku alebo ho C-APDU vôbec nemusí obsahovať. Hlavička má veľkosť 4 B a každá jej položka má 1 B. Jednotlivé bajty hlavičky sú znázornené na obr. 1.7.[6]



Obr. 1.7: Štruktúra hlavičky C-APDU.

CLA definuje typ inštrukcie. INS zase definuje kód použitej inštrukcie, ktorý reprezentuje konkrétny príkaz. Parametre P1 a P2 sa používajú ako doplnkové informácie o príkaze, ktorý je definovaný v INS. [6]

Pokiaľ je v C-APDU zahrnuté aj telo, to môže obsahovať 3 rôzne časti: Lc, DATA a Le. Použité však nemusia byť všetky. V časti DATA sú uložené dáta, ktoré sú potrebné k vykonaniu požadovaného príkazu. Pole Lc udáva veľkosť dátovej časti v poli DATA. Pole Le zase definuje veľkosť dátovej časti, ktorá sa očakáva v odpovedi na daný príkaz. Na obr. 1.8 je znázornené, v akej kombinácií môžu byť tieto 3 polia použité.[5]



Obr. 1.8: Možné štruktúry tela C-APDU.

Pokiaľ ide o možnosť 1, vtedy príkaz na kartu síce neobsahuje žiadne riadiace dáta, no v odpovedi karty sa dáta očakávajú. Pri možnosti 2 je to zase naopak, príkaz dáta obsahuje, ale žiadne dáta sa v odpovedi od karty neočákavajú. V poslednej tretej možnosti ide o kombináciu týchto dvoch možností, a teda aj príkaz obsahuje riadiace dáta, a taktiež sa dáta očakávajú v odpovedi od karty. V prípade, že C-APDU neobsahuje telo vôbec, samozrejme dáta neobsahuje ani príkaz a neočakáva sa to ani od odpovede.[5]

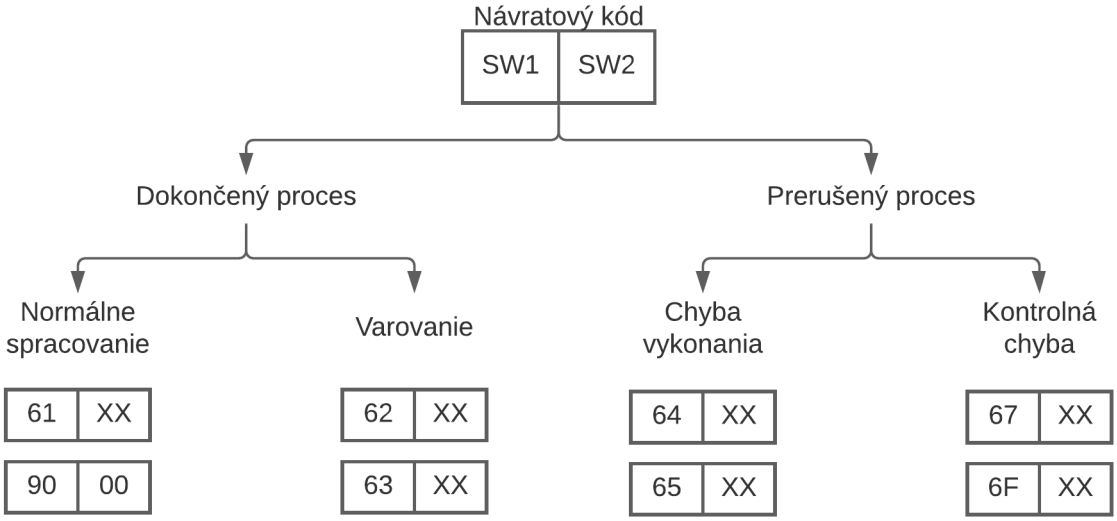
Štandardne môžu mať dáta v poli DATA veľkosť 0 - 255 B. Je však možné použiť rozšírený príkaz C-APDU, ktorý podporuje veľkosť poľa DATA až 1 - 65 535 B. S tým súvisí aj veľkosť poľa Lc. Tá môže byť o veľkost 0 B, 1 B alebo 3 B. Pokiaľ je jeho veľkosť 0 B, C-APDU neobsahuje žiadne dáta. Ak je veľkosť 1 B, vtedy C-APDU obsahuje dáta o veľkosť 1 - 255 B. Ak je ale veľkosť 3 B, dáta môžu byť až do veľkosti práve 65 535 B a ide o rozšírený C-APDU. Obdobne môže byť veľkosť poľa Le 0 B, 1 B, 2 B, alebo 3 B.

Štruktúra R-APDU

R-APDU sa taktiež delí na 2 časti, z ktorých je len 1 časť povinná. Tu sú ale tieto 2 časti telo (nepovinná časť) a prípona. Nepovinné telo obsahuje iba dátovú časť. Veľkosť tejto dátovej časti je určená v príkaze predchádzajúcom túto odpoveď. Prípona obsahuje 2 polia: SW1 a SW2. Tieto 2 polia spolu udávajú návratový kód, ktorý je reakciou na predchádzajúci príkaz. Existuje veľa návratových kódov, ktoré sú štandardizované a ľahko dohľadateľné. Avšak treba poznamenať, že v praxi sa často používajú aj neštandardizované kódy pri rôznych aplikáciách. Asi najznámejší a taktiež jediný návratový kód, ktorý je zahrnutý aj medzi štandardizovanými aj neštandardizovanými kódmi je kód „9000“. Označuje úspešné spracovanie. Štruktúra R-APDU je znázornená na obr. 1.9.[5]

Obr. 1.9: Štruktúra R-APDU.

Návratové kódy môžu byť rôzneho typu. Rozdelenie návratových kódov je znázornené na obr. 1.10.[5]



Obr. 1.10: Rozdelenie návratových kódov v R-APDU.

1.4 NFC

NFC je bezkontaktná komunikačná technológia, ktorá slúži na komunikáciu na krátke vzdialenosti. Bola vyvinutá pri vývoji technológií indukčnej väzby pri RFID a technológií čipových kariet. Najskôr bola štandardizovaná v normách ECMA-340 a ECMA-352, neskôr však boli prijaté aj v normách ISO/IEC 18092 a ISO/IEC 21481.[8]

NFC Forum je združenie, ktoré zoskupuje organizácie, ktorých predmetom záujmu je technológia NFC. Toto združenie sa stará o vytváranie špecifikácií pre dátové formáty, protokoly a pod. pre technológiu NFC.[8]

NFC zariadenia môžeme rozdeliť na aktívne a pasívne. Aktívne sú tie, ktoré majú vlastný zdroj energie, naopak pasívne tie, ktoré nemajú. Tie získavajú energiu pomocou elektromagnetickej indukcie za pomoci iniciátora. Do tejto druhej kategórie spadajú aj bezkontaktné čipové karty.

NFC pracuje v troch prevádzkových režimoch:

- režim peer-to-peer (rovný s rovným),
- režim čítačka / zapisovač,
- režim emulácie karty.[8]

1.4.1 Peer-to-peer

Tento režim prevádzky NFC umožňuje rovnocennú komunikáciu medzi dvoma zariadeniami NFC. Pre tento režim komunikácie je špecifikovaných niekoľko protokolov, medzi nimi napríklad LLCP.[8]

1.4.2 Režim čítačka / zapisovač

V tomto režime zariadenie NFC dokáže získať prístup k pasívnym NFC štítkom (viac o NFC štítkoch v časti 1.4.4). Taktiež v tomto režime môže interagovať s bezdotykovými RFID transpondérmi a čipovými kartami. Vďaka tomuto režimu sú NFC zariadenia schopné súčinnosti s čipovými kartami a taktiež so starými infraštruktúrami značiek RFID.[8]

1.4.3 Režim emulácie karty

Ako už z názvu vyplíva, v tomto režime je NFC zariadenie schopné emulovať bezkontaktnú čipovú kartu. To znamená, že v podstate dokáže nahradiť bežnú bezkontaktnú čipovú kartu a jej funkcionalitu. Dosiahne sa tým to, že NFC zraiadenie je schopné súčinnosti so starými infraštruktúrami čítačiek RFID.[8]

1.4.4 NFC štítky

Princíp NFC štítkov spočíva v tom, že po interakcii štítku s NFC zariadením dôjde na danom zariadení k spusteniu určitej akcie. Ako príklad môže poslúžiť využitie NFC štítku ako súčasť reklamy. Po tom, čo sa smartfón so zariadením NFC priblíži k tomuto štítku, ten v smartfóne spustí napríklad akciu otvorenia webovej stránky.[8]

1.4.5 Komunikácia NFC rozhrania s čipovou kartou

Pokiaľ zariadenie NFC pracuje v peer-to-peer móde, komunikácia je založená na základe štandardizácie ISO/IEC 18092. V módoch čítačka/zapisovač alebo emulácie karty je však NFC zariadenie schopné komunikácie aj podľa štandardizácie ISO/IEC 14443. Čo sa týka režimu emulácie karty, tam je zjavné, že pokiaľ má zariadenie s rozhraním NFC nahrádzať čipovú kartu, musí byť schopné komunikácie podľa tejto štandardizácie. [9]

Bezkontaktné čipové karty sú schopné komunikácie podľa štandardizácie ISO-/IEC 14443. Pokiaľ teda ide o režim čítačky/zapisovača, kde NFC štítok predstavuje karta, nakoľko chceme dosiahnúť komunikáciu medzi zariadením s podporou NFC rozhrania a čipovej karty, je nutné túto komunikáciu zabezpečovať práve na základe tejto štandardizácie. [9]

1.5 Programovanie čipových kariet

Termín „otvorená platforma“ označuje operačné systémy čipových kariet, ktoré umožňujú ich užívateľom načítať do nich vlastné navrhnuté aplikácie a programy bez toho, aby do tohto procesu bol zahrnutý výrobca. Existuje veľa takýchto platform pre čipové karty. Medzi najznámejšie patria:

- Java Card,
- Multos Card,
- Basic Card.[5]

Pre účel tejto práce je použitá práve platforma Java Card.

1.5.1 Java Card

Platforma Java Card podporuje programovací jazyk JavaCard, čo je vlastne obmedzená Java. Narozdiel od „klasickej“ Java nepodporuje napríklad dátový typ `integer`, vykonávanie viac ako jedného vlákna alebo polia s viac ako jednou dimenziou.[10]

Architektúra appletu

Základná časť architektúry je applet. Ten musí byť zahrnutý v určitom balíku, pričom balík môže obsahovať viacero appletov. Tento balík si teda môžeme predstaviť ako akýsi kontajner čipovej karty, ktorý môže obsahovať viacero komponentov príslušnej aplikácie.[10]

Applet na svoje fungovanie potrebuje, aby mohol využívať aj iné dostupné balíčky. Tieto balíčky treba importovať. Základné balíčky, ktoré pre svoju funkčnosť potrebuje, sú:

- `javacard.framework.APDU` - obsahuje podporu APDU, ktoré sú nevyhnutné na komunikáciu,
- `javacard.framework.Applet` - z tohto balíčka musí byť nami vytváraný applet odvodený, obsahuje totiž metódy ako sú napríklad `install` alebo `process` (vysvetlené ďalej),
- `javacard.framework.ISOException` - umožňuje appletu spracovávať a vyvolávať výnimky.[10]

Najdôležitejšie metódy, ktoré musí applet obsahovať sú vyššie spomínané `install` a `process`. Metóda `install` slúži na registráciu appletu do JCRE. Registrácia appletu je nevyhnutná na to, aby vôbec mohol byť voliteľný terminálom. Metóda je volaná na konci načítania aplikácie. Platí, že pokiaľ by nebola táto metóda volaná, aplikácia síce bude načítaná na čipovú kartu, avšak nebude sa s ňou dať komunikovať terminálom. V tejto metóde zvyknú byť deklarované tie dátové položky, ktoré

chceme aby boli vykonané iba raz počas doby aktivity appletu (napríklad kryptografické kľúče).[10]

Metóda `process` je volaná vtedy, keď je na kartu poslaný APDU. Ten je jej vstupným parametrom. Táto metóda je považovaná za hlavnú funkciu appletu, nakoľko slúži ako rozhranie medzi terminálom a funkciami appletu. Slúži na analýzu APDU a vykonáva tú funkciu, ktorú APDU vyžaduje.[10]

Okrem týchto dvoch metód sú často využívané aj metódy `select` a `deselect`. Metóda `select` je používaná na informovanie appletu o tom, že bol vybraný, metóda `deselect` naopak o tom, že jeho výber bol zrušený. Tieto metódy sa samozrejme môžu využiť aj na to, aby vykonali akcie, ktoré chceme, aby sa vykonali iba raz, a to na začiatku aktivity appletu resp. na konci (príklad, ako ich je možné využiť bude uvedený v praktickej časti v časti 2.1.5).[10]

1.6 Programovanie Android aplikácií

Existuje viacero softvérov na tvorbu android aplikácií. Asi najznámejším je Android studio, ktorý je použitý aj v rámci tejto práce. Android Studio je oficiálne IDE pre vývoj aplikácií pre Android, ktoré je založené na IntelliJ IDEA.

1.6.1 Základné metódy Android aplikácie

Základ celej aplikácie tvoria aktivity. Aktivita je hlavná trieda, ktorá je pre užívateľa viditeľná po spustení aplikácie. Je to vlastne nástroj, pomocou ktorého užívateľ aplikáciu ovláda. Aplikácia môže samozrejme pozostávať z viacerých aktivít.[12]

Metóda `onCreate` umožňuje inicializovať aktivitu, je volaná pri prvom vytvorení aktivity. V nej by sa mali vykonať všetky bežné nastavenia, ako je zobrazenie prvkov aktivity a definovanie ich funkcií. Metóda `onResume` sa volá, keď začne aktivita interagovať s používateľom. Metóda `onPause` sa volá, keď daná aktivita už nie je v popredí a nahradí ju iná aktivita.[12]

1.6.2 Použitie rozhrania NFC v Android aplikáciách

Základná trieda, ktorá je využívaná pri používaní NFC rozhrania je trieda `NfcAdapter`. Objekt tejto triedy predstavuje NFC adaptér, ktorý dané zariadenie obsahuje.[11]

Je potrebné si zadefinovať pojem zámer. Zámer je istá dátová štruktúra. Tá môže niesť informáciu o tom, aká operácia sa má vykonať. Taktiež ale môže hovoriť o tom, aká udalosť v systéme nastala a vyžaduje si obsluhu. Ak je do adaptéra prijímaný nejaký NFC zámer, vyvolá sa metóda `onNewIntent`. Tento zámer je argumentom tejto metódy. Je nutné, aby vývojar túto metódu upravil tak, aby tento zámer

spracovala na základe toho, aké zámery má aplikácia spracovávať. Metóda sa vyvolá na základe akéhokoľvek zámeru, preto je potrebné nejakým spôsobom vyfiltrovať tie zámery, ktoré nijako nesúvisia s NFC. To vieme dosiahnúť napríklad tým, že na zažiatku zadefinujeme podmienku. Po tom, ako sa overí, že ide o NFC zámer resp. nejaký konkrétny NFC zámer, z tohto zámeru sa získa inštancia triedy `Tag`.[13], [11]

Predmetom tejto práce je komunikácia s bezkontaktnou čipovou kartou. Komunikácia s ňou podlieha špecifikácií ISO/IEC 14443. Takúto komunikáciu dokáže zabezpečiť trieda `IsoDep`. Keď karta príde do kontaktu s rozhraním NFC a teda s NFC adaptérom, ten zachytí zámer od karty. Vyššie opísaným spôsobom vieme získať `Tag`. Potom je potrebné už iba tento `Tag` upraviť, príklad je uvedený vo výpise 1.1.[14]

Výpis 1.1: Príklad vytvorenia a zadefinovania inštancie triedy `IsoDep`.[14]

```
static IsoDep myTag;
myTag = IsoDep.get(tag);
```

Trieda `IsoDep` obsahuje metódu `transceive`, ktorá dokáže zaslať karte prostredníctvom NFC rozhrania vo forme bajtového poľa APDU príkaz.[14]

1.7 Využitie certifikátov pri komunikácií

Pokiaľ chce účastník komunikovať v systéme, kde prebieha komunikácia na základe digitálneho podpisu, musí tento účastník disponovať verejným kľúčom VK a súkromným kľúčom SK. Pokiaľ má takýto systém fungovať, je potrebné, aby existovala certifikačná autorita CA. Každý účastník musí poznať jej verejný kľúč VK_{CA}. Pre účastníka je potrebné, aby si vyžiadal svoj certifikát od certifikačnej autority. Certifikát je vlastne certifikačnou autoritou digitálne podpísaná správa o tom, že daný účastník A disponuje verejným kľúčom VK_A. Ak teda dvaja účastníci nadväzujú komunikáciu, pomocou známeho VK_{CA} si prijatý certifikát vedia overiť. Pokiaľ je certifikát platný, znamená to aj že verejný kľúč ktorý certifikát obsahuje je platný a môžu na základe tohto kľúča ďalej overovať platnosť podpisu správy.[19]

Hlavné informácie, ktoré musí certifikát obsahovať sú identita účastníka a jeho verejný kľúč. V praxi ale obsahujú aj ďalšie informácie, ako sú napríklad verzia formátu certifikátu, sériové číslo certifikátu, algoritmus podpisu alebo platnosť certifikátu. To, ako budú vyzerať súkromné a verejné kľúče závisí od toho, aký algoritmus je na podpisovanie používaný.[19]

1.7.1 Štandard X.509

Tento štandard popisuje formát certifikátov, ich parametre a metódy, ktorými kontroluju platnosť certifikátov. Štandard definuje takýto formát certifikátu:

1. Verzia - bližšie špecifikuje formát certifikátu.
2. Sériové číslo - jedinečné číslo, priradzuje ho certifikačná autorita.
3. Špecifikácia použitého podpisu - špecifikuje algoritmus, ktorý je použitý k podpisovaniu a ďalšie potrebné parametre.
4. Certifikačná autorita - identifikátor certifikačnej autority.
5. Doba platnosti - obsahuje údaje, od kedy je certifikát platný a do kedy.
6. Subjekt - meno vlastníka certifikátu.
7. Verejný kľúč subjektu - kľúč subjektu a algoritmus.[18][19]

Príklad toho, ako môže vyzerať certifikát je znázornený na obr. 1.11.

```
        Version: 1
   SerialNumber: 123456789
       IssuerDN: CN=CAName
     Start Date: Fri Jan 1 00:00:00 CEST 2021
     Final Date: Thu Jan 1 00:00:00 CEST 2026
      SubjectDN: CN=SubjName
     Public Key: EC Public Key [9b:6b:60:12:db:53:65:7b:51:af:ca:26:c0:97:58:5e:b6:29:c7:2c]
      X: c8e93cfa5546d2e894f22ef5613d83af8a1b7f90ef2f08dd
      Y: 74c43d087d16ef72c96afff9523cf6f3785286c511e82388

Signature Algorithm: ECDSA
      Signature: 3036021900d03e566cf72b45902446acc46021f5
                 c0cb1107737c02e46202190090ac9485544b3668
                 a7da5bbef7b4db3f325d688d7ad9a8f1
```

Obr. 1.11: Príklad certifikátu.

1.8 ECDSA

ECDSA je kryptograficky zabezpečená schéma digitálneho podpisu. Založená je na kryptografii eliptickej krivky. Je to variant podpisovej schémy DSA. Princíp ECDSA spočíva v tom, že sa spolieha na problém diskrétneho logaritmu eliptickej krivky. Výhodou je, že ECDSA kľúče a podpisy sú na rozdiel od napríklad RSA kratšie pre rovnakú úroveň zabezpečenia. V praxi to znamená, že rovnakú silu zabezpečenia, akú ponúka 256 bitový podpis ECDSA ponúka 3072 bitový podpis RSA.[17]

Eliptická krivka je definovaná ako množina bodov, ktoré vyhovujú rovnici

$$y^2 = x^3 + ax + b$$

kde x, y, a, b sú reálne čísla. Ak platí, že

$$4a^3 + 27b^2 \neq 0$$

potom sa dá eliptická krivka využiť k vytvoreniu grupy.[17]

1.8.1 Eliptické krivky

Existuje veľa zaužívaných kriviek používaných pre techniku ECDSA, ktoré sú štandardizované. Jednou z najznámejších spoločností, ktoré štandardizovali eliptické krivky je NIST. Tá ďalej delí krivky na eliptické krivky podľa štandardizácie SECP a SECT. Eliptické krivky SECP sú také, ktoré sú definované nad prvočíselným poľom F_p. Naopak, eliptické krivky SECT sú tie, ktoré sú definované nad poľom F_{2^m}. Ďalej sa tieto krivky delia na k-krivky a r-krivky.[15]

V praxi sa v kryptografii v súvislosti s eliptickými krivkami používajú krivky nad prvočíselným poľom F_p alebo poľom F_{2^m}. Od toho, o aké pole ide závisí aj to, aké systémové parametre krivky sa používajú. Pokiaľ ide o krivku nad prvočíselným poľom, tak systémové parametre sú (p, a, b, G, n, h). Ak naopak ide o krivku nad poľom F_{2^m}, potom sú parametre (m, f(x), a, b, G, n, h). V rámci tejto práce bude použitá eliptická krivka nad prvočíselným poľom F_p. Význam jej parametrov je nasledovný:

- p – prvočíslo, ktorým je definované pole,
- Parametre a, b – definícia eliptickej krivky,
- Bod G – generátor - základný bod na krivke,
- n – rád bodu G (najmenšie číslo, ktorým keď vynásobíme G, dostaneme bod v nekonečne),
- h – kofaktor – voliteľný parameter.[15]

Krivka secp192k1

Pre túto prácu bola zvolená krivka secp192k1. Krivka secp192k1 je eliptická krivka nad 192 bitovým prvočíselným poľom F_p. Patrí ku krivkám štandardizovaných spoločnosťou NIST a ako už z názvu vyplíva, spadá pod štandardizáciu SECP. Má nasledujúce parametre:

- p = `FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFEE37`
- a = `00000000 00000000 00000000 00000000 00000000 00000000`
- b = `00000000 00000000 00000000 00000000 00000000 00000003`
- G = `04 DB4FF10E C057E9AE 26B07D02 80B7F434 1DA5D1B1 EAE06C7D 9B2F2F6D 9C5628A7 844163D0 15BE8634 4082AA88 D95E2F9D`
- n = `FFFFFFFF FFFFFFFF FFFFFFFE 26F2FC17 0F69466A 74DEFD8D`
- h = `01`[15]

1.8.2 Princíp podpisovania podľa ECDSA

Tento algoritmus potrebuje na vstupe disponovať správou, ktorú chceme podpísať a súkromný kľúč. Výstupom algoritmu je podpis vstupnej správy, ktorý pozostáva z čísel r a s.

Algoritmus pozostáva z troch častí:

1. Stanovenie kľúčového páru (súkromný a verejný kľúč) vyhovujúci algoritmu ECDSA.
2. Generovanie podpisu správy.
3. Verifikácia prijatého podpisu.

Prvým krokom je stanovenie verejného a privátneho podpisového kľúča. Pri použití ECDSA musí platiť, že

$$VK = SK * G$$

Ďalším krokom je určenie prvej polovice podpisu označovanej ako r. Na to je najprv potrebné vypočítať heš správy m. Takto dostaneme heš

$$h = hash(m)$$

Ďalej je potrebné vybrať náhodné číslo k z rozsahu (1, n-1). Keď je známe číslo k, je možné pokračovať ďalej podľa vzorca

$$(x_1, y_1) = k * G$$

Z toho následne vyplýva prvá časť podpisu podľa vzorca

$$r = x_1 \mod n$$

Pokiaľ by došlo k situácií, kedy r = 0, je potrebné vygenerovať nové číslo k. Je to z toho dôvodu, že keby sa r rovnalo nule, podpis by nebol závislý od súkromného kľúča.

Druhá časť podpisu s sa vypočíta zo vzťahu

$$s = k^{-1} * (h + r * SK) \mod n$$

Obdobne ako pri r, aj pri s platí, že sa nesmie rovnať nule. Keď je podpis správy vygenerovaný ako dvojica (r, s), môže podpísanú správu odosielateľ poslať.

Pri overení podpisu na strane prijímateľa je najprv potrebné skontrolovať, či prijaté r a s z podpisu sú z intervalu (1, n-1). Pokiaľ áno, je potrebné vypočítať heš h prijatej správy obdobne ako pri tvorbe podpisu. Ďalej prijímateľ musí vypočítať hodnotu w, pričom

$$w = s^{-1} \mod n$$

Následne je potrebné vypočítať hodnoty

$$u_1 = h * w \mod n$$

a

$$u_2 = r * w \mod n$$

Na základe týchto hodnôt vieme určiť

$$X = u_1 G + u_2 S K_A$$

a

$$v = x_1 \mod n$$

Ak

$$v = r$$

, potom bol podpis overený a správu môže prijímateľ považovať za autentickú.[16], [17]

1.9 Hešovacie funkcie

Jednosmerná funkcia je taká funkcia, kde pre vzor sa dá vypočítať jeho obraz, no naopak pre obraz nie je možné spätne vypočítať jeho vzor. Síce nie je dokázané, či takéto funkcie vôbec naozaj existujú, no v kryptografi sa používajú, pričom sa verí, že takúto vlastnosť majú. Nazývajú sa hešovacie funkcie.[19]

Princíp hešovacích funkcií spočíva v tom, že vzoru, ktorý je ľubovoľnej dĺžky sa priradí jeho obraz pevne stanovenej dĺžky. V závislosti na tom v akom pomere sú dĺžky vzoru a obrazu, sa delia hešovacie funkcie na:

- kompresné - vzor je kratší ako obraz,
- ekvivalentné - vzor je rovnakej dĺžky ako obraz,
- expanzné - vzor je dlhší ako obraz.[19]

Hešovacie funkcie môžu byť rôznych dĺžok. Často sa používaju hlavne tie, ktorých dĺžka hešu je 128 bitov až 512 bitov. Konkrétne ide o hešovacie funkcie MD5 (128 b), SHA-1 (160 b), SHA-2 (224/256/384/512 b).

V rámci kryptografie sa hešovacie funkcie využívajú najmä v autentizačných kryptosystémoch. Ich účel spočíva v tom, že správam vieme vytvárať ich reprezentantov konštantnej dĺžky. Správa V sa označuje ako vzor hešovacej funkcie H a má dĺžku L. Obrazom vzoru a teda výstupom hešovacej funkcie je heš h, ktorý je vopred stanovenej dĺžky N.[19]

Hešovacia funkcia musí spĺňať 3 základné požiadavky:

- odolnosť voči získaniu vzoru - pre heš h musí byť prakticky nemožné získať vzor V - pravdepodobnosť získania vzoru je závislá na dĺžke hešu, pričom pravdepodobnosť je 2^{-N},
- odolnosť voči modifikácií vzoru - nesmie byť možné, aby na základe hešu $h = H(V_1)$ sa dal nájsť vzor V_2 pričom sa tieto dva vzory nesmú rovnať,
- odolnosť voči kolíziam - musí byť nemožné, aby heše 2 rôznych vzorov mali rovnakú hodnotu.[19]

1.9.1 SHA-1

SHA-1 je kryptografická hešovacia funkcia, ktorej výstupom je heš o dĺžke 160 bitov.

Prvým krokom pri tejto hešovacej funkcii je vyplnenie správy. Správa totiž môže mať rôznu dĺžku, avšak na to, aby mohla byť spracovaná hešovacou funkciou SHA-1 musí mať dĺžku, ktorej hodnota je násobkom 512 bitov. Realizácia výplne správ prebieha nasledovne. Máme pôvodnú správu prezentovanú v binárnej forme:

```
01100001 01100010 01100011 01100100 01100101
```

Za túto správu sa vloží binárna jednotka:

```
01100001 01100010 01100011 01100100 01100101 1
```

Ďalej sa miesta za jednotkou vyplnia nulami tak, aby v rezerve ostalo ešte posledných 64 bitov chýbajúcich do dĺžky 512 bitov. Po prevedení do hexadecimálnej formy to vyzerá nasledovne:

```
61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000
```

V posledných 64 bitoch bude uložená veľkosť, ktorú mala pôvodná správa. V tomto prípade je to konkrétne hodnota 28 v hexadecimálnej sústave. Vyplnená správa tak bude vyzerať takto:

```
61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000028
```
[20]

Vyplnená správa tak teraz obsahuje `16*n` slov.

Ďalej je pre SHA-1 definovaných päť 32-bitových slov `H0 - H4`:

`H0 = 67452301`

`H1 = EFCDAB89`

`H2 = 98BADCFE`

`H3 = 10325476`

`H4 = C3D2E1F0`

Vyplnená správa sa rozdelí na `n` blokov, ktoré sú označované ako `M(1) - M(n)`. Následne sa postupuje takto:

1. Každá `M(i)` sa ďalej rozdelí na 16 slov `W(0) - W(15)`.
2. Pre `t = 16 až 79` sa vykoná:

$$W(t) = ROL^1(W(t-3)XORW(t-8)XORW(t-14)XORW(t-16))$$

3. Nech `A = H0`, `B = H1`, `C = H2`, `D = H3`, `E = H4`.
4. Pre `t = 0 až 79` sa vykoná:

$$TEMP = ROL^5(A) + f(t; B, C, D) + E + W(t) + K(t);$$

$$E = D; D = C; C = ROL^{30}(B); B = A; A = TEMP;$$

5. Nech `H0 = H0 + A`, `H1 = H1 + B`, `H2 = H2 + C`, `H3 = H3 + D`, `H4 = H4 + E`.

Po tom, ako sa spracuje `M(n)` máme 160 bitový reťazec.

2 Výsledky študentskej práce

2.1 Návrh riešenia a implementácia

Cieľom tejto práce je vytvoriť dve aplikácie, ktoré demonštrujú myšlienku použitia bezkontaktnej čipovej karty ako skrýše pre geokešing.[1] Komunikáciu je potrebné realizovať pomocou navrhnutého protokolu bližšie špecifikovaného v priloženej špecifikácií zadania.

Návrh riešenia sa dá rozdeliť do návrhu dvoch aplikácií. V prvom rade je potrebné navrhnúť applet pre bezkontaktnú kartu. Na to, aby mohla byť karta použitá ako skrýš je taktiež ďalej potrebné navrhnúť hernú aplikáciu pre smartfón, pomocou ktorej môže užívateľ komunikovať s kartou resp. získať odmenu za nájdenie danej skrýše. Cieľom tejto práce je na základe získaných znalostí popísaných v teoretickej časti vytvoriť funkčný systém dvoch aplikácií, ktoré by sa dali aplikovať pre geokešing.

2.1.1 Princíp fungovania aplikácií

Vychádzame z toho, že karta predstavuje v rámci geokešingu skrýšu. To, ako skrýša zareaguje po naviazaní spojenia s hráčovým smartfónom závisí od toho, o akú formu odmeny má hráč záujem alebo lepšie povedané akú hru si hráč zvolí. Hľadanie fyzickej skrýše v rámci geokešing funguje na príncípe, že hráč sa po nájdení tejto skrýše zapíše do logovacej knihy a taktiež si túto skrýšu označí ako nájdenú na hernom serveri. Hľadať už nájdenú skrýšu nemá pre hráča zmysel. Tento princíp je potrebné zachovat aj pri týchto aplikáciach. Hráčovi teda nebude umožnené opätovne nadviazať spojenie so skrýšou a získať za jej nájdenie odmenu opäť. Na hráčovej strane je teda potrebné viesť zoznam už nájdených skrýš. Pokiaľ aplikácia rozpozná, že skrýš ku ktorej sa hráč chce pripojiť ním už v minulosti bola nájdená, spojenie ukončí.

Podľa špecifikácie zadania musí byť možné organizovať 2 typy hier:

- Putovanie virtuálnych predmetov,
- Zberateľstvo virtuálnych predmetov.

Pre zabezpečenie organizácie putovných predmetov je potrebné, aby applet karty podporoval ukladanie a vymazanie identifikátora virtuálneho predmetu. Hráčová aplikácia a karta si teda vymieňajú identifikátory predmetov. Pokiaľ hráč nedisponuje žiadným putovným predmetom, alebo pokiaľ v skrýši nie je uložený žiadny virtuálny predmet, identifikátor v hernej aplikácií resp. na karte má nulovú hodnotu. Tu je potrebné definovať, čo sa má diať v prípade, že identifikátor má nulovú hodnotu. Môžu nastať 4 prípady s rôznymi následkami:

1. Hráč aj skrýša disponujú putovným predmetom - do skrýše sa uloží identifikátor od hráča a hráč už nebude vlastniť tento predmet, u hráča sa uloží identifikátor predmetu, ktorý bol uložený v skrýši (následne bude putovať do inej skrýše).
2. Hráč disponuje putovným predmetom, skrýša nie - do skrýše sa uloží identifikátor putovného predmetu od hráča, hráč ale na oplátku nedostane žiaden predmet, ktorý by mohol ďalej putovať.
3. Skrýša disponuje putovným predmetom, hráč nie - u hráča sa uloží tento identifikátor a predmet môže ďalej putovať, skrýša ale ostane prázdna a bude čakať na iného hráča, ktorý do nej putovný predmet vloží.
4. Ani hráč ani skrýša nedisponujú putovným predmetom - nenastane žiadna výmena.

Ďalším typom hry, ktorá je podporovaná, je zberateľstvo virtuálnych predmetov. Tu je možnosť voľby na základe toho, aké hry podporuje herný server. Pre účel tejto práce boli definované možnosti pre hru skladačky a zberateľstva obrázkov prírodných krás. Tento typ hier nepotrebuje prijímať žiadne dáta s identifikátormi od karty. Od karty potrebuje iba potvrdenie, že daná skrýš bola nájdená. Keď hráčová alikácia toto potvrdenie získa, je už v réžii samotnej aplikácie, aby vykonala potrebný krok v rámci hry (odhalenie nového obrázka prírodných krás, odhalenie časti skladačky).

2.1.2 Návrh formátu prenášaných hodnôt

Na základe špecifikácie zadania je potrebné, aby hodnoty, ktoré sa budú v rámci správ prenášať, sa prenášali pomocou formátu AVP. Každá hodnota sa bude prenášať v trojici. Táto trojica je označovaná ako Y, pričom Y = (A, L, V). A je typ hodnoty, L je celková dĺžka trojice a V je samotná hodnota. Na základe toho bolo potrebné zvoliť si číslovanie typov prenášaných hodnôt. Navrhnuté číslovanie je vidieť v tab. 2.1.

Ďalej je tiež potrebné určiť, akej dĺžky budú niektoré hodnoty:

- ID skrýše - veľkosť 12 B (prvé dva bajty sú znaky GC),
- ID hráča - veľkosť 22 B (prvý bajt je znak P),
- ID putovného predmetu - veľkosť 22 B (prvé dva bajty sú znaky TG),
- Náhodé čísla - veľkosť 5 B.

2.1.3 Návrh komunikácie

V tomto návrhu je potrebné, aby smartfón komunikoval s kartou v režime čítačka / zapisovač. Smartfón resp. jeho NFC rozhranie bude teda komunikovať na základe štandardizácie ISO/IEC 14443.

Tab. 2.1: Použité premenné v applete karty

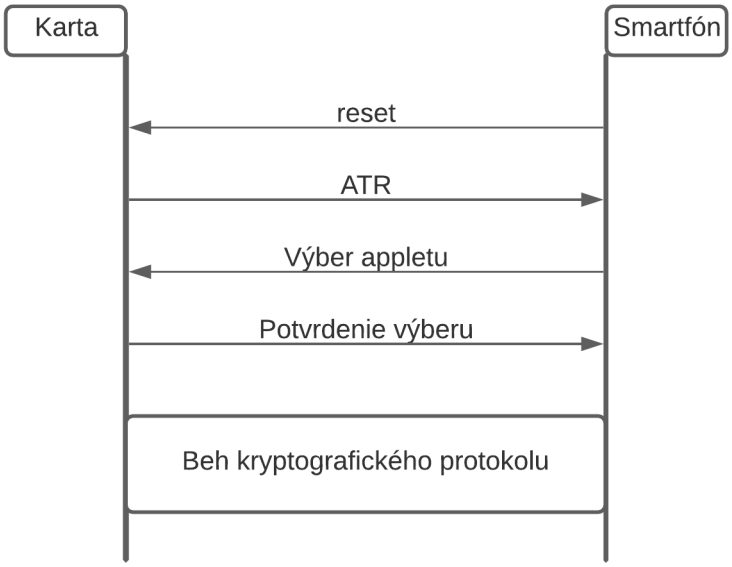
Priradené číslo	Hodnota	Označenie
1	ID skrýše	S
2	ID hráča	H
3	ID putovného predmetu hráča	U_H
4	ID putovného predmetu v skrýši	U_S
5	certifikát	CRT
6	verejný kľúč skrýše	VK_S
7	verejný kľúč hráča	VK_H
8	podpis skrýše	P_S
9	podpis hráča	P_H
10	náhodné číslo skrýše	R_S
11	náhodné číslo hráča	R_H
12	správa skrýše	Z_S
13	správa hráča	Z_H
14	poradové číslo nálezcu skrýše	N
15	verejný kľúč certifikačnej autority	VK_{CA}
16	podpis certifikačnej autority	P_{CA}

Na zabezpečenie komunikácie bude použitý transportný protokol T=1 predstavený v časti 1.3.1. Dáta budú prenášané prostredníctvom APDU. Nakoľko ale dáta, ktoré budú prenášané sú relatívne veľké (väčšie ako 255 B), používať sa bude rošírený formát APDU. Návrh priebehu komunikácie je znázornený na obr. 2.1.

Ako je vidieť, po spustení aplikácie na smartfóne je potrebné najprv samozrejme priložiť smartfón ku skrýši, teda ku karte. Smartfón si následne resetom vyžiada od karty ATR. Po prijatí ATR od karty je potrebné smartfónom vybrať applet karty, ktorý obsahuje logiku pre beh kryptografického porotokolu pre komunikáciu aplikácie hráča a skrýše. Karta obsahuje iba tento jediný applet. Smartfón teda zašle AID daného appletu na kartu a karta aktivuje tento applet. V prípade úspešného výberu appletu karta odošle odpoveď oznamujúcu túto skutočnosť. Prijatie kladnej odpovede smartfónom na jeho strane iniciuje začiatok behu kryptografického protokolu.

Komunikáciu medzi smartfónom a kartou môžeme rozdeliť na 2 časti. V prvej časti dôjde k nadviazaniu spojenia medzi kartou a smartfónom a výber appletu, druhá časť predstavuje už samostatný kryptografický protokol.

V prvej časti dôjde k zaslaniu dvoch príkazov zo smartfónu na kartu a na nich nadväzujúce dve odpovede z karty na smartfón. Tieto príkazy je potrebné zadefino-



Obr. 2.1: Navrhovaný priebeh komunikácie medzi smartfónom a kartou.

vať ako APDU príkazy podľa štruktúry popísanej v časti 1.3.4.

Navrhnutý príkaz pre výber appletu je znázornený na obr. 2.2. Obsahuje aj dátovú časť, ktorá tvorí AID daného appletu.

Zaslanie AID

00	A4	04	00	08	AB	CD	EF	FE	DC	12	34	56	00

Obr. 2.2: Navrhovaná štruktúra príkazu C-APDU posielaného na kartu pre výber appletu.

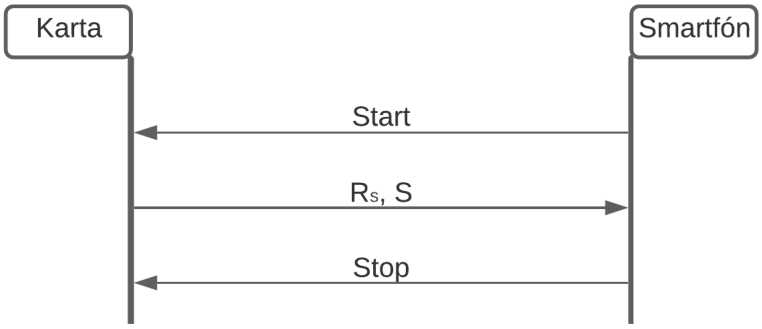
Ako odpoveď sa očakáva iba potvrdenie a teda nie je potrebné aby odpoveď obsahovala dáta. Bude teda obsahovať iba polia SW1 a SW2 s hodnotami `90 00`.

Pri samotnom kryptografickom protokole môžu nastať 2 prípady:

1. Hráč už danú skrýšu niekedy našiel.
2. Hráč ešte danú skrýšu nikdy nenašiel.

Komunikácia pre prvý prípad je znázornená na obr. 2.3.

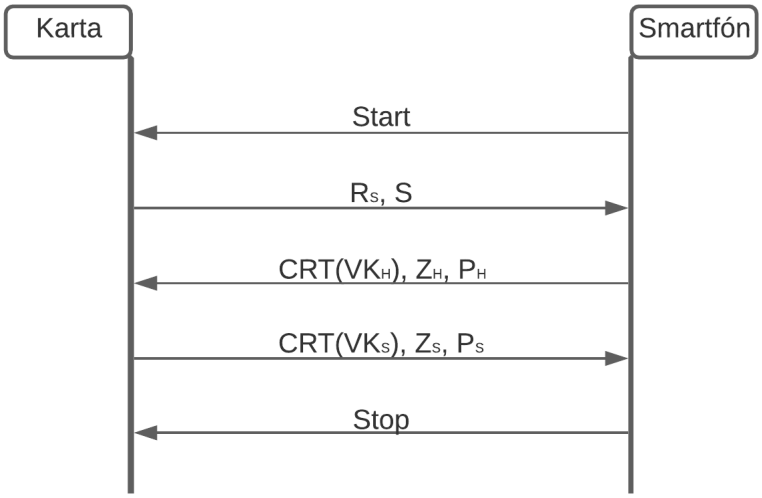
Po tom, čo sa na karte vyberie applet a smartfón o tom dostane potvrdenie, z hráčovej aplikácie sa odošle karte príkaz Start. Karta na neho zareaguje tak, že obratom pošle hráčovej aplikácií hodnoty R_s a S. S predstavuje 12 bajtový indentifikátor



Obr. 2.3: Priebeh komunikácie v prípade, že danú skrýš už hráč niekedy v minulosti našiel.

skrýše. Hráčová aplikácia tento identifikátor porovná s identifikátormi skrýš, ktoré už hráč našiel. Pokiaľ nastane zhoda, znamená to, že hráč už túto skrýš niekedy v minulosti našiel. A práve to sa v tomto prípade stane. Hráč už teda nemôže získať žiadnú odmenu za nájdenie skrýše a celá komunikácia s kartou sa ukončí zaslaním príkazu Stop na kartu.

Na obr. 2.4 je naopak znázornené, ako prebieha komunikácia, ak je skrýš pre hráča nová.



Obr. 2.4: Priebeh komunikácie v prípade, že danú skrýš hráč ešte nikdy predtým nenašiel.

Obdobne, ako v predchádzajúcom prípade, po zvolení appletu odosiela hráčová aplikácia karte príkaz Start a karta obratom posiela hráčovej aplikácií hodnoty R_s a S. V tomto prípade ale pri overení identifikátora S v zozname už nájdených skrýší nenastala zhoda. Znamená to teda, že skrýša ešte nikdy predtým nebola hráčom nájdená. Pre smartfón to znamená, že môže začať s výmenou identifikátora putovného predmetu. Dátová časť APDU, ktoré hráčová aplikácia odošle karte, bude obsahovať 3 časti:

- certifikát hráča,
- správu hráča,
- hráčov podpis správy.

Certifikát je položka, ktorú má hráč vo svojej aplikácií bezpečne uloženú. Karte ju posiela, aby bola schopná pomocou verejného kľúča, ktorý je súčasťou certifikátu, overiť podpis hráčovej správy. Samotná správa obsahuje štyri položky:

1. Náhodné číslo hráča R_H, ktoré hráčová aplikácia vygeneruje pre tento beh protokolu.
2. Náhodné číslo skrýše R_S, ktoré skrýš vygenerovala pre tento beh protokolu a predtým ho zaslala hráčovej aplikácií spolu s identifikátorom.
3. Identifikátor skrýše S, ktorý skrýša predtým odoslala hráčovej aplikácii.
4. Identifikátor putovného predmetu hráča U_H, ktorý hráčska aplikácia chce vložiť do skrýše.

Za samotnou správou ešte nasleduje podpis tejto správy.

Skrýša po prijatí C-APDU najprv overí hráčov certifikát. Ak je v poriadku, verejný kľúč hráča použije na overenie podpisu prijatej správy. Po overení skontroluje, či sa prijaté náhodné číslo R_S zhoduje s tým, ktoré predtým vygenerovalo a poslalo smartfónu a taktiež, či sa prijatý identifikátor S zhoduje s jej identifikátorom. Identifikátor putovného predmetu bude ďalej vymenený s tým, ktorý sa momentálne v skrýši nachádza. Odpoveď skrýše R-APDU na prijaté C-APDU bude taktiež pozostávať s troch častí:

- certifikát skrýše,
- správu skrýše,
- podpis správy skrýše.

Pre certifikát platí to isté, ako u hráča, a teda že obsahuje verejný kľúč skrýše. Samotná správa ale v tomto prípade bude obsahovať o niečo viac položiek:

1. Náhodné číslo skrýše R_S, ktoré skrýš vygenerovala pre tento beh protokolu.
2. Náhodné číslo hráča R_H, ktoré hráčová aplikácia vygenerovala pre tento beh protokolu a predtým ho zaslala skrýši.
3. Identifikátor skrýše S.
4. Identifikátor hráča H, ktorý skrýša pozná z hráčového certifikátu.

5. Poradové číslo N, ktoré určuje, koľký v poradí je hráč medzi hráčmi, ktorí už túto skrýš našli.
6. Identifikátor putovného predmetu skrýše U_S, ktorý skrýš predáva hráčovi pre ďalšie putovanie.
7. Identifikátor putovného predmetu hráča U_H, ktorý hráčska aplikácia vložila do skrýše.

Za správou taktiež nasleduje jej podpis.

Keď hráč príjme C-APDU, obdobne ako skrýš overí certifikát skrýše a ak je v poriadku, verejným kľúčom overí podpis správy. Ak je podpis korektný, môže spracovávať prijatú správu. V prvom rade overí náhodné čísla R_S, R_H, identifikátor skrýše S, identifikátor hráča H a identifikátor putovného predmetu U_H, ktorý hráčska aplikácia uložila do skrýše. Identifikátor putovného predmetu U_S si teraz uloží ako identifikátor putovného predmetu hráča a pri nájdení inej skrýše už bude prezentovaný ako U_H. Hráčska aplikácia považuje touto odpoveďou proces nájdenia skrýše za ukončeý a skrýši posiela príkaz Stop.

2.1.4 Návrh certifikátov pre komunikujúce strany

Certifikáty, ktoré boli pridelené hráčovi a skrýši sa riadia formátom štandardu X.509. Pre obe komunikujúce strany bola najprv vygenerovaná dvojica kľúčov (súkromný a verejný). Na základe ich verejného kľúča a identifikátora bol certifikačnou autoritou vygenerovaný certifikát a následne aj podpísaný.

V tejto práci bol pre podpisovanie správ zvolený algoritmus `SHA1withECDSA`. To znamená, že rovnakým algoritmom sa bude riadiť aj podpisovanie certifikátov. Kľúče musia byť generované tak, aby zodpovedali parametrom eliptickej krivky secp192k1.

Pre generovanie certifikátov bol naprogramovaný samostatný program v jazyku Java. Ten na základe balíčka `java.security.cert.X509Certificate` a knižnice `org.bouncycastle` generuje certifikáty. Ako súkromný a verejný kľúč certifikačnej autority bola vygenerovaná dvojica:

Súkromný kľúč:
`CCD6B28D3DD8DB96FBC9EE049DDAF22BC7C4C2733DCF11AD`
Verejný kľúč:
`0486EF1D163701BCC0EC9628EB8F129447A3F2EB5B73C405FF`
`C7877BEE8AF1EC4C36A12AF6FFD9CB52A416C183904E052B`

Jej súkromným kľúčom sa certifikáty podpisujú, verejný kľúč získavajú spolu s certifikátom obaja držitelia certifikátov a overujú nim certifikát toho druhého.

Dvojice kľúčov pre hráča a pre skrýšu boli vygenerované nasledovné:

Skrýš - súkromný kľúč:
`38476478E1945F4CBF7F82F82A5B28C23653E03E6E722D3A`
Skrýš - verejný kľúč:
`04C8E93CFA5546D2E894F22EF5613D83AF8A1B7F90EF2F08DD`
`74C43D087D16EF72C96AFFF9523CF6F3785286C511E82388`

Hráč - súkromný kľúč:
`C81E1F9A19493A2829C08F367FB7D873D9CCCD3CFDAB6768`
Hráč - verejný kľúč:
`0445640C90E183BF663ACF6D344A8D654F9A940823D42229DE`
`7FCD7634C7954D625F11CDF0ACD5EB1EDB0C3FC49EF2413B`

Celý certifikát skrýše je vidieť na obr. 2.5. Ako je vidieť, jeho kľúč sa zhoduje s vyššie uvedenou hodnotou a jeho ID je `GC0123456789`.

```
          Version: 1
     SerialNumber: 1621430261675
         IssuerDN: CN=Geocaching
       Start Date: Wed May 19 15:17:41 CEST 2021
       Final Date: Tue May 19 15:17:41 CEST 2026
        SubjectDN: CN=GC0123456789
       Public Key: EC Public Key [9b:6b:60:12:db:53:65:7b:51:af:ca:26:c0:97:58:5e:b6:29:c7:2c]
        X: c8e93cfa5546d2e894f22ef5613d83af8a1b7f90ef2f08dd
        Y: 74c43d087d16ef72c96afff9523cf6f3785286c511e82388

Signature Algorithm: ECDSA
          Signature: 3036021900d03e566cf72b45902446acc46021f5
                     c0cb1107737c02e46202190090ac9485544b3668
                     a7da5bbef7b4db3f325d688d7ad9a8f1
```

Obr. 2.5: Certifikát skrýše.

Ďalej certifikát hráča je vidieť na obr. 2.6. Ako je vidieť, jeho kľúč sa zhoduje s vyššie uvedenou hodnotou a jeho ID je `P998765432100123456789`.

Certifikáty sa medzi kartou a smartfónom musia v rámci APDU samozrejme prenášať ako bajtové pole. Preto výstupom generátora certifikátov je taktiež certifikát vo formáte DER. Sú to v podstate informácie certifikátu zakódované do binárnej formy (resp. hexadecimálnej). V takomto poli bajtov si už následne applet karty resp. aplikácia smartfónu vie nájsť informácie z certifikátu, ktoré potrebuje. Pre ukážku, certifikáty v DER formáte je vidieť na obr. 2.7.

```
            Version: 1
       SerialNumber: 1621430830696
           IssuerDN: CN=Geocaching
         Start Date: Wed May 19 15:27:10 CEST 2021
         Final Date: Tue May 19 15:27:10 CEST 2026
          SubjectDN: CN=P998765432100123456789
         Public Key: EC Public Key [13:da:1b:8a:1a:98:76:c4:49:fc:8d:ee:6c:fc:b8:5a:c2:c7:8f:c7]
          X: 45640c90e183bf663acf6d344a8d654f9a940823d42229de
          Y: 7fcd7634c7954d625f11cdf0acd5eb1edb0c3fc49ef2413b

Signature Algorithm: ECDSA
          Signature: 30350218086e3d74fe8c591c0d7fa7415f99db5f
                     37687cc7aa7299a7021900cce8c8bb0903393c60
                     81ef5a284a40b775bbf01d1001a840
```

Obr. 2.6: Certifikát hráča.

Certifikát skrýše

```
30 82 01 A3 30 82 01 59  02 06 01 79 84 C6 57 AB
30 09 06 07 2A 86 48 CE  3D 04 01 30 15 31 13 30
11 06 03 55 04 03 13 0A  47 65 6F 63 61 63 68 69
6E 67 30 1E 17 0D 32 31  30 35 31 39 31 33 31 37
34 31 5A 17 0D 32 36 30  35 31 39 31 33 31 37 34
31 5A 30 17 31 15 30 13  06 03 55 04 03 13 0C 47
43 30 31 32 33 34 35 36  37 38 39 30 81 F3 30 81
BC 06 07 2A 86 48 CE 3D  02 01 30 81 B0 02 01 01
30 24 06 07 2A 86 48 CE  3D 01 01 02 19 00 FF FF
FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
FF FE FF FF EE 37 30 34  04 18 00 00 00 00 00 00
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00 00 04 18 00 00 00 00  00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00  00 00 00 03 04 31 04 DB
4F F1 0E C0 57 E9 AE 26  B0 7D 02 80 B7 F4 34 1D
A5 D1 B1 EA E0 6C 7D 9B  2F 2F 6D 9C 56 28 A7 84
41 63 D0 15 BE 86 34 40  82 AA 88 D9 5E 2F 9D 02
19 00 FF FF FF FF FF FF  FF FF FF FF FF FE 26 F2
FC 17 0F 69 46 6A 74 DE  FD 8D 02 01 01 03 32 00
04 C8 E9 3C FA 55 46 D2  E8 94 F2 2E F5 61 3D 83
AF 8A 1B 7F 90 EF 2F 08  DD 74 C4 3D 08 7D 16 EF
72 C9 6A FF F9 52 3C F6  F3 78 52 86 C5 11 E8 23
88 30 09 06 07 2A 86 48  CE 3D 04 01 03 39 00 30
36 02 19 00 D0 3E 56 6C  F7 2B 45 90 24 46 AC C4
60 21 F5 C0 CB 11 07 73  7C 02 E4 62 02 19 00 90
AC 94 85 54 4B 36 68 A7  DA 5B BE F7 B4 DB 3F 32
5D 68 8D 7A D9 A8 F1
```

Certifikát hráča

```
30 82 01 AC 30 82 01 63  02 06 01 79 84 CF 06 68
30 09 06 07 2A 86 48 CE  3D 04 01 30 15 31 13 30
11 06 03 55 04 03 13 0A  47 65 6F 63 61 63 68 69
6E 67 30 1E 17 0D 32 31  30 35 31 39 31 33 32 37
31 30 5A 17 0D 32 36 30  35 31 39 31 33 32 37 31
30 5A 30 21 31 1F 30 1D  06 03 55 04 03 13 16 50
39 39 38 37 36 35 34 33  32 31 30 30 31 32 33 34
35 36 37 38 39 30 81 F3  30 81 BC 06 07 2A 86 48
CE 3D 02 01 30 81 B0 02  01 01 30 24 06 07 2A 86
48 CE 3D 01 01 02 19 00  FF FF FF FF FF FF FF FF
FF FF FF FE FF FF EE 37  30 34 04 18 00 00 00 00
00 00 00 00 00 00 00 00  00 00 00 00 04 18 00 00
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00 00 00 00 00 03 04 31  04 DB 4F F1 0E C0 57 E9
AE 26 B0 7D 02 80 B7 F4  34 1D A5 D1 B1 EA E0 6C
7D 9B 2F 2F 6D 9C 56 28  A7 84 41 63 D0 15 BE 86
9D 02 19 00 FF FF FF FF  26 F2 FC 17 0F 69 46 6A
74 DE FD 8D 02 01 01 03  32 00 04 45 64 0C 90 E1
83 BF 66 3A CF 6D 34 4A  8D 65 4F 9A 94 08 23 D4
22 29 DE 7F CD 76 34 C7  95 4D 62 5F 11 CD F0 AC
D5 EB 1E DB 0C 3F C4 9E  F2 41 3B 30 09 06 07 2A
86 48 CE 3D 04 01 03 38  00 30 35 02 18 08 6E 3D
74 FE 8C 59 1C 0D 7F A7  41 5F 99 DB 5F 37 68 7C
C7 AA 72 99 A7 02 19 00  CC E8 C8 BB 09 03 39 3C
60 81 EF 5A 28 4A 40 B7  75 BB F0 1D 10 01 A8 40
```

Obr. 2.7: Certifikáty hráča a skrýše v DER formáte.

2.1.5 Návrh aplikácie pre bezkontaktnú čipovú kartu a implementácia

Pre aplikáciu karty s názvom `GeocachingApp` je navrhnutý kód, ktorý vychádza z predchádzajúceho návrhu komunikácie v 2.1.3.

Základné metódy appletu karty, ktoré musí applet obsahovať sú z hľadiska ich významu popísané v časti 1.5.1. Tieto metódy `install`, `select` a `deselect`, okrem ich primárneho účelu nerobia nič špeciálne. Metóda `process` vyhodnocuje prijaté APDU. Podľa toho, či prijaté APDU obsahuje zadefinovanú požadovanú hodnotu

CLA, metóda process buď pokračuje ďalej v spracovaní, alebo odošle smartfónu chybovú R-APDU. Ak pokračuje v spracovaní ďalej, overuje či hodnota INS je jednou z trojice zadefinovaných požadovaných hodnôt INS. Ak áno, vyhodnocuje ktorá presne hodnota zo zadefinovanej trojice to je a na základe toho ďalej postupuje. Použité atribúty pre definovanie podporovaných hodnôt CLA a INS sú nasledovné:

- final static byte Geocaching_CLA = 0x90 - hodnota CLA, ktoré musí obsahovať APDU, aby ho karta mohla ďalej spracovávať,
- final static byte START = 0x01 - hodnota INS, ktoré musí obsahovať APDU príkaz pre začatie kryptografického protokolu,
- final static byte STOP = 0x02 - hodnota INS, ktoré musí obsahovať APDU príkaz pre ukončenie kryptografického protokolu (či už z dôvodu úspešného prebehnutia protokolu alebo vyhodnotenia, že hráč už danú skrýšu našiel),
- final static byte GAME = 0x03 - hodnota INS, ktoré musí obsahovať APDU príkaz pre prijatie dát zo strany smartfónu a následné odoslanie ďalších dát z karty.

Z definovaných hodnôt INS vyplíva, že aplikácia musí byť v prvom rade schopná zareagovať na 3 rôzne požiadavky od smartfónu. Konkrétne ide o požiadavky na začatie kryptografického protokolu, jeho ukončenie a výmenu potrebných údajov.

Ak sa pri vyhodnocovaní INS zistí, že ide o hodnotu START, zaháji sa kryptografický protokol. K tejto časti protokolu patrí iba jedna zadefinovaná metóda:

- private void startProtocol - pripraví a odošle smartfónu hodnoty R_S a S.

Pokiaľ sa vyhodnotí, že CLA sa rovná hodnote STOP, zaháji sa ukončenie kryptografického protokolu. Ak teda karta príjme APDU s takýmto CLA, znamená to, že smartfón vyhodnotil identifikátor skrýše za už známy a teda táto skrýša už v minulosti bola hráčom nájdená. K ukončeniu protokolu nebola navrhnutá žiadna špeciálna metóda. Na karte je zadefinovaná premenná boolean stop, ktorá automaticky pri každom nadviazaní spojenia so smartfónom nadobúda hodnotu false. Hodnota tejto premennej sa kontroluje po každom prijatí APDU ešte pred jeho analýzou. Ak je hodnota false, môže sa ďalej pokračovať s analýzou. Naopak, ak je true, blokuje sa akákoľvek ďalšia práca s prijatým APDU. Zmenu tejto hodnoty na true zabezpečuje práve prijatie APDU s parametrom CLA sa rovným hodnote STOP.

Nakoniec ešte môže karta vyhodnotiť, že CLA sa rovná hodnote GAME. V tomto prípade začína hlavná časť celého kryptografického protokolu. Tá sa dá rozdeliť do dvoch krokov, konkrétne na prijatie hodnôt zo smartfónu a na odoslanie hodnôt z karty do smartfónu. V prvom kroku boli navrhnuté nasledujúce metódy:

- private void receive - príjma C-APDU v rozšírenom formáte (väčšia ako

255 B),

- `private short getCRThFromData` - z dátovej časti APDU vyberie certifikát hráča a uloží si ho,
- `private boolean verifyCRT` - z uloženého certifikátu hráča oddelí informačné dáta certifikátu od podpisu certifikačnej autority a následne overením podpisu zhodnotí, či je certifikát v poriadku,
- `private void getIDFromCert` - je volaná, ak metóda `verifyCRT` má kladný výsledok. Z certifikátu si vyberie ID hráča a uloží si ho,
- `private void getPubKFromCert` - je volaná, ak metóda `verifyCRT` má kladný výsledok. Z certifikátu si vyberie verejný kľúč hráča a uloží si ho,
- `private short getZhFromData` - z dátovej časti APDU vyberie celú podpísanú správu hráča a uloží si ju,
- `private short getPhFromData` - z dátovej časti APDU vyberie podpis správy hráča a uloží si ho,
- `private boolean verifyPh` - kontroluje autentickosť predtým uloženej správy tým, že overí získaný podpis správy,
- `private boolean getAndCheckSubdataFromZh` - je volaná, ak metóda `verifyPh` má kladný výsledok. Ďalej analyzuje predtým vybranú správu hráča. Vyberie z nej jednotlivé hodnoty premenných. Hodnoty R_H a U_H si uloží. Hodnoty R_S a S porovná s tými, ktoré má karta zadefinované a ak sú v poriadku, vracia kladnú hodnotu a protokol môže ďalej pokračovať.

V druhom kroku boli navrhnuté nasledujúce metódy:

- `private short sign` - slúži na podpisovanie správ, je volaná metódou `preparePsToSend`,
- `private void assignCurveParametersPubK` - je volaná pri vytváraní objektu triedy `ECPublicKey` a teda verejného kľúča, priradzuje kľúču parametre požadovanej eliptickej krivky,
- `private void assignCurveParametersPrivK` - je volaná pri vytváraní objektu triedy `ECPrivateKey` a teda súkromného kľúča, priradzuje kľúču parametre požadovanej eliptickej krivky,
- `private void createCacheKeyPair` - vytvára súkromný a verejný kľúč skrýše podľa zadefinovaných hodnôt týchto kľúčov, vytvára teda objekt triedy `ECPrivateKey` a `ECPublicKey`,
- `private void createPlayerPubKey` - vytvára verejný kľúč hráča podľa zistenej hodnoty kľúča z certifikátu, vytvára teda objekt triedy `ECPublicKey`,
- `private void createCAPubKey` - vytvára verejný kľúč certifikačnej autority podľa získanej hodnoty tohto kľúča, vytvára teda objekt triedy `ECPublicKey`,
- `private byte[] valueToAVPformat` - upravuje prenášané hodnoty do smartfónu do formátu AVP.

Ďalšie pomocné metódy appletu karty sú:

- `private short prepareDataToSend` - je volaná, ak metóda `getAndCheckSubdataFromZh` má kladný výsledok. Zahajuje prípravu dát na odoslanie smartfónu. Vyvoláva ďalšie metódy pripravujúce jednotlivé časti posielaných dát,
- `private short prepareCertToSend` - je volaná, metódou `prepareDataToSend`. Pridá certifikát skrýše k dátam na odoslanie,
- `private short prepareZsToSend` - je volaná, metódou `prepareDataToSend`. Pridá jednotlivé hodnoty tvoriace správu skrýše k dátam na odoslanie (R_S, R_H, S, H, N, U_S, U_H). Zároveň zvyšuje počet nájdení skrýše a taktiež zabezpečuje výmenu identifikátora putovného predmetu, ktorý skrýša obsahuje,
- `private short preparePsToSend` - je volaná, metódou `prepareDataToSend`. Najprv podpíše odosielanú správu a následne podpis pridá k dátam na odoslanie,
- `private void send` - odosiela R-APDU v rozšírenom formáte (väčšia ako 255 B).

Nakoľko hodnoty prenášané medzi smartfónom a kartou sa prenášajú vo formáte AVP, každá hodnota musí mať svoj identifikátor. V zdrojovom kóde karty sú zadefinované nasledovne:

- `final static byte CACHE_ID = 0x01` - prenášaná hodnota je ID skrýše,
- `final static byte GAMER_ID = 0x02` - prenášaná hodnota je ID hráča,
- `final static byte GAMER_TRAV_ITEM_ID = 0x03` - prenášaná hodnota je ID putovného predmetu z hráčovej aplikácie,
- `final static byte CACHE_TRAV_ITEM_ID = 0x04` - prenášaná hodnota je ID putovného predmetu zo skrýše,
- `final static byte CERT = 0x05` - prenášaná hodnota je certifikát,
- `final static byte CACHE_PUB_KEY = 0x06` - prenášaná hodnota je verejný kľúč skrýše,
- `final static byte GAMER_PUB_KEY = 0x07` - prenášaná hodnota je verejný kľúč hráča,
- `final static byte CACHE_SIGN = 0x08` - prenášaná hodnota je podpis skrýše,
- `final static byte GAMER_SIGN = 0x09` - prenášaná hodnota je podpis hráča,
- `final static byte CACHE_RNDM_NUMB = 0x10` - prenášaná hodnota je náhodné číslo, ktoré vygenerovala skrýša,
- `final static byte GAMER_RNDM_NUMB = 0x11` - prenášaná hodnota je náhodné číslo, ktoré vygeneroval hráč,
- `final static byte CACHE_CONF_MSG = 0x12` - prenášaná hodnota je správa skrýše,
- `final static byte GAMER_MSG = 0x13` - prenášaná hodnota je správa hráča,

- `final static byte DISC_ORDER = 0x14` - prenášaná hodnota je poradové číslo hráča ako nálezcu skrýše,
- `final static byte CA_PUB_KEY = 0x15` - prenášaná hodnota je verejný kľúč certifikačnej autority,
- `final static byte CA_SIGN = 0x16` - prenášaná hodnota je podpis certifikačnej autority.

Aplikácia pre kartu je písaná v jazyku JavaCard v `NetBeans IDE`. Prácu s APDU zabezpečuje knižnica `javacard.framework.*`.

2.1.6 Návrh aplikácie pre smartfón a implementácia

Aplikácia pre smartfón je navrhnutá pre čo najjednoduchšie ovládanie. Taktiež je navrhnutá na základe návrhu komunikácie v 2.1.3. Zadávanie APDU je automatizované, nie je potrebné zadávanie celého APDU užívateľom. Aplikácia musí byť schopná:

- nadviazať spojenie s kartou odoslaním AID,
- zahájiť kryptografikcý protokol,
- ukončiť kryptografický protokol (buď z dôvodu vyhodnotenia skrýše ako už nájdenej v minulosti alebo úspešného prebehnutia protokolu).

Na základe toho bude musieť aplikácia obsahovať 4 základné metódy:

1. `private boolean sendAID` - odosiela karte AID,
2. `private boolean sendStartAPDU` - odosiela karte príkaz pre začatie kryptografického protokolu, zároveň príjma od karty odpoveď (náhodné číslo a identifikátor skrýše),
3. `private boolean sendPlayerData` -odosiela karte svoje dáta potrebné k behu protokolu, zároveň príjma od karty ako odpoveď jej dáta potrebné k behu protokolu,
4. `private boolean sendStopAPDU` - odosiela karte príkaz, ktorý ukončí beh kryptografického protokolu.

Všetky 4 metódy vracajú hodnotu typu `boolean`. Smartfón tak na základe toho môže reagovať na skutočnosť, či karta spracovala zaslané údaje v poriadku resp. či údaje, ktoré jej smartfón odoslal sú v poriadku. K prvej metóde nie sú priradené žiadne prídavné metódy. S druhou metódou sú ďalej spojené tieto metódy:

- `private void storeRsS` - ukladá do smartfónu náhodné číslo od skrýše a jej identifikátor pre ďalší beh protokolu,
- `private boolean sendPlayerData` - v zozname už nájdených skrýš overuje, či sa medzi nimi nenachádza identifikátor tejto skrýše.

Metódu `private boolean sendPlayerData` môžeme z hľadiska zložitosti označiť za najkomplikovanejšiu. Dá sa rozdeliť na 2 časti. V prvom rade je potrebné odoslať potrebné dáta karte. V druhej časti je následne potrebné spracovať dáta odpovede od karty. S touto metódou sú spojené ďalšie:

- `private int prepareCertToSend` - skopíruje dáta hráčovho certifikátu na začiatok bajtového poľa, ktoré bude odoslané karte,
- `private int prepareZhToSend` - do bajtového poľa, ktoré bude poslané karte, postupne za certifikát kopíruje všetky potrebné hodnoty tvoriace správu hráča,
- `private int preparePsToSend` - volá metódu `sign` na podpísanie správy a pridá ju do bajtového poľa na odoslanie za správu,
- `private int sign` - podpíše hráčovú správu súkromným kľúčom hráča,
- `private boolean storeCacheData` - po prijatí dát zo skrýše zaháji ich ukladanie a overovanie, volá nasledujúce metódy,
- `private int getCRTsFromData` - z odpovede skrýše vyberie certifikát skrýše a uloží ho do premennej `CRTs`,
- `private boolean verifyCRT` - overí certifikát skrýše uložený v premennej `CRTs`,
- `private int getZsFromData` - z odpovede skrýše vyberie časť so správou skrýše a uloží ju do premennej `Zs`,
- `rivate int getPsFromData` - z odpovede skrýše vyberie poslednú časť s podpisom skrýše a uloží ju do premennej `Ps`,
- `private boolean verifyPs` - overí podpis skrýše správy uloženej v premennej `Zs`,
- `private boolean getAndCheckSubdataFromZs` - ďalej analyzuje premennú `Zs`. Rozdeľuje ju na samostatné hodnoty, ktoré skrýša odoslala (R_S, R_H, S, H, N, U_S, U_H) a ukladá ich do premenných. S výnimkou premenných N a U_S u všetkých overuje ich správnosť. Ak sú v poriadku, uloží si hodnotu N a U_S. Hodnota U_S je identifikátor putovného predmetu od skrýše, ktorý teraz bude patriť hráčovi a tak ju uloží do premennej U_H,
- `private void getIDFromCert` - z certifikátu skrýše vyberie identifikátor skrýše (použitý na overenie),
- `private void getPubKFromCert` - z certifikátu skrýše vyberie verejný kľúč skrýše (použitý na podpisu `Ps`),
- `public void getReward` - po úspešnom priebehu kryptografického protokolu v aplikácii zobrazí vyskakovacie okno pre hru "Zbierka virtuálnych predmetov".

Aplikácia samozrejme obsahuje aj iné metódy, ktoré boli rozdelené do kategórií. Prvou kategóriou sú metódy spojené s odosielaním a prijímaním APDU:

- `public static byte[] transceives` - odosiela bajtové pole, ktoré predsta-

vuje C-APDU. Výsledkom metódy je taktiež bajtové pole, tentoraz ale predstavujúce dáta prijatej odpovede,
- `private byte[] receiveData` - z prijatého C-APDU vyberá dátovú časť (bez návratového kódu),
- `public static boolean checkCorrectAnswer` - overuje, či je odpoveď od karty v poriadku. Kontroluje, či návratový kód SW má hodnotu `9000`.

Ďalšou kategóriou sú metódy vytvárajúce kľúče potrebné k podpisovaniu správ a overovaniu podpisov. Pre vytvorenie kľúčov je potrebné mať ich hodnoty vo forme bajtového poľa. Hodnoty hráčových kľúčov sú známe už pred začiatkom behu protokolu. Taktiež verejný kľúč certifikačnej autority. Ten hráč obdržal spolu so svojím certifikátom od certifikačnej autority. Verejný kľúč skrýše bol vybraný z certifikátu skrýše behom protokolu. K vytvoreniu kľúčov sú potrebné dalšie parametre kľúča. Kedže sa v tomto riešení používa eliptická krivka secp192k1, práve jej parametre sú uložené v aplikácii a použijú sa pri tvorbe kľúčov. Kľúče použité na podpisovanie alebo overenie sú objekty triedy `ECPrivateKey` resp. `ECPublicKey`. Obe sú vytvorené pomocou objektu triedy `ECPrivateKeySpec` resp. `ECPublicKeySpec`. V oboch prípadoch je jedným z atribútov parametre krivky secp192k1. Ďalším atribútom je v prípade verejného kľúča bod na krivke (objekt triedy `ECPoint`), ktorý je daný bajtovým poľom pre uchovávanie kľúča. Ten je prevedený na 2 hodnoty typu `BigInteger`, ktoré určujú tento bod. U súkromného kľúča je bajtové pole taktiež prevedené na hodnotu typu `BigInteger`, tu je to ale iba jedno číslo ktoré udáva parameter `s`. To je ďalší atribút v prípade súkromného kľúča.
- `private void createPlayerPrivKe` - vytvára súkromný kľúč hráča k podpisovaniu správ (objekt triedy `ECPrivateKey`),
- `private void createCAPubKey` - vytvára verejný kľúč certifikačnej autority k overeniu certifikátu (objekt triedy `ECPublicKey`),
- `private void createCachePubKey` - vytvára verejný kľúč skrýše k overeniu podpisu správy (objekt triedy `ECPublicKey`).

Samozrejme aplikácia obsahuje aj ďalšie metódy. Patria k nim napríklad také, ktore zobrazujú chybové hlášky počas behu aplikácie (napríklad o nesprávnych prijatých dátach a pod.). Taktiež sú tam pomocné metódy. Tie napríklad upravujú prenášané hodnoty do formátu AVP, alebo vkladajú hodnoty typu `integer` do bajtového poľa.

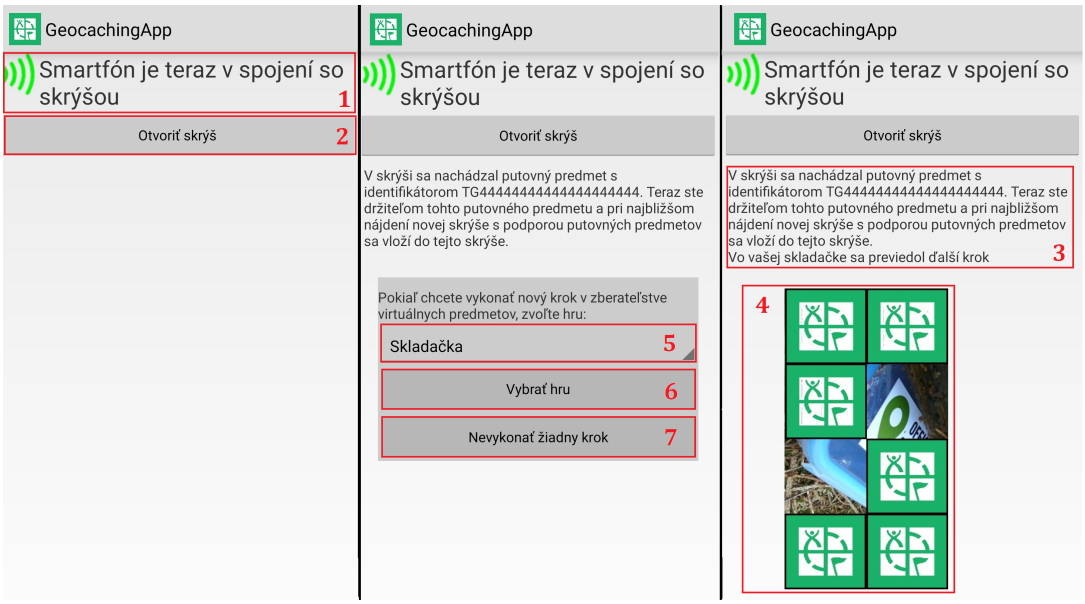
Taktiež z hľadiska android aplikácie ako takej sú potrebné metódy:
- `onCreate` - viď. 1.6.1,
- `onResume` - viď. 1.6.1,
- `onPause` - viď. 1.6.1,

- `onNewIntent` - pokiaľ je do adaptéra prijímaný nejaký zámer, vyvolá metódu `resolveIntent`, kde argument je tento zámer,
- `resolveIntent` - vykonáva identifikáciu zaznamenaného zámeru, vytvára `Tag`.

Užívateľské rozhranie tvorí jedná hlavná stránka, ktorá obsahuje:

1. Indikátor pripojenia karty. Pokiaľ aplikácia detekuje kartu, ikona sa rozsvieti na zeleno a zobrazí sa príslušná správa.
2. Tlačidlo „Otvoriť skrýšu“. Odošle AID aplikácie na kartu a ďalej prevádza kroky kryptografického protokolu.
3. Textové pole pre výpis, zobrazuje informácie o predaní putovného predmetu a vykonaní kroku v zberateľstve virtuálnych predmetov.
4. Obrázkove pole pre zobrazenie odmeny za prevedený krok v hre.
5. Zoznam hier. Po jeho rozkliknutí sa zobrazia na výber 2 hry.
6. Tlačidlo „Vybrať hru“. Stlačením tlačidla sa prevedie krok v získavaní virtuálnych predmetov podľa toho, aká hra zo zoznamu je zvolená.
7. Tlačidlo „Nevykonať žiadny krok“. Stlačením tlačidla zruší výber hier.

Tieto prvky sú zobrazené na obr. 2.8, ktorý zobrazuje užívateľské rozhranie aplikácie.



Obr. 2.8: Užívateľské rozhranie aplikácie pre smartfón a jeho prvky.

2.2 Test aplikácií

Testy sú rozdelené do dvoch skupín. Je totiž potrebné otestovať ako aplikáciu na karte, tak aj aplikáciu pre smartfón.

2.2.1 Test aplikácie na karte

Pre účel tohto testu bola použitá voľne dostupná aplikácia PyApduTool, ďalej len testovacia aplikácia. Táto aplikácia je schopná odosielať javakarte zadané APDU príkazy. Priebeh komunikácie pritom vypisuje do terminálu. Test sa dá rozdeliť do niekoľkých scenárov.

Test zaslania chybného AID karte

Prevedenie testu: Testovacia aplikácia po prijatí ATR od karty zaslala APDU príkaz pre výber appletu s chybným AID.
Výsledok: Terminál korektne zaslal APDU príkaz s chybným AID karte (zámerne vložená chyba zvýraznená na obr. 2.9). Ako je ďalej z výpisu vidieť, odpoveďou je návratový kód `6A82`, ktorý vraví o nenájdenom súbore, čo je v poriadku nakoľko applet so zaslaným AID na karte neexistuje.

```
Send: 00 A4 04 00 08 AA CD EF FE DC 12 34 56 00
Recv: 6A 82
```

Obr. 2.9: Výpis komunikácie z terminálu - zaslanie chybného AID.

Test zaslania chybných dát správy karte

Prevedenie testu: Karta v prvom kroku po zahájení odosiela smartfónu svoje náhodné číslo a svoj identifikátor. V ďalšom kroku má prijať okrem iných hodnôt obratom aj tieto dve, ktoré porovnáva s pôvodnými. Testovacia aplikácia v tomto scenári odošle upravený APDU príkaz, kde bola pozmenená hodnota náhodného čísla.
Výsledok: Po výbere appletu (korektnom, keďže odpoveď bola `9000`) terminál korektne zaslal APDU príkaz pre začatie kryptografického protokolu. Ako odpoveď karta odoslala náhodné číslo a svoj identifikátor. Do ďalšieho APDU príkazu, ktorý karta obratom prijala bola vložená chyba do položky náhodného čísla skrýše (pôvodná hodnota náhodného čísla a zámerne vložená chyba sú zvýraznené na obr. 2.10). Ako je ďalej z výpisu vidieť, odpoveďou je návratový kód `6984`, ktorý vraví o tom, že údaje v dátovom poli sú neplatné. Chybový návratový kód ale v skutočnosti

nehovorí o tom, že porovnávané náhodné čísla sa nezhodujú. V behu aplikácii na karte totiž porovnávaniu čísel predchádza overenie podpisu správy. Nakoľko bola zmenená hodnota náhodného čísla, bola zmenená aj samotná správa. Táto zmena bola ale zámerne urobená až po podpísaní pôvodnej (korektnej) správy. Ak by bola hodnota zmenená ešte pred podpísaním, chybový návratový kód by bol totožný, no vznikol by už na základe porovnávania náhodných čísel.

```
Send: 90 01 00 00
Recv: 10 07 89 E3 F9 52 0C 01 0E 47 43 30 31 32 33 34 35 36 37 38 39 90 00
Time used: 10.000 ms
Send: 90 03 00 00 00 02 20 05 01 B3 30 82 01 AC 30 82 01 63 02 06 01 79 84 CF 06 68 30 09 06 07 2A 86 48 CE 3D 04 01 30 15
31 13 30 11 06 03 55 04 03 13 0A 47 65 6F 63 61 63 68 69 6E 67 30 1E 17 0D 32 31 30 35 31 39 31 33 32 37 31 30 5A 17 0D 32
36 30 35 31 39 31 33 32 37 31 30 5A 30 21 31 1F 30 1D 06 03 55 04 03 13 16 50 39 39 38 37 36 35 34 33 32 31 30 30 31 32 33
34 35 36 37 38 39 30 81 F3 30 81 BC 06 07 2A 86 48 CE 3D 02 01 30 81 B0 02 01 01 30 24 06 07 2A 86 48 CE 3D 01 01 02 19 00
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FE FF FF EE 37 30 34 04 18 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 04 18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 04 31 04 DB 4F F1 0E
C0 57 E9 AE 26 B0 7D 02 80 B7 F4 34 1D A5 D1 B1 EA E0 6C 7D 9B 2F 2F 6D 9C 56 28 A7 84 41 63 D0 15 BE 86 34 40 82 AA 88 D9
5E 2F 9D 02 19 00 FF FF FF FF FF FF FF FF FF FF FF FE 26 F2 FC 17 0F 69 46 6A 74 DE FD 8D 02 01 01 03 32 00 04 45 64 0C 90 E1 83
BF 66 3A CF 6D 34 4A 8D 65 4F 9A 94 08 23 D4 22 29 DE 7F CD 76 34 C7 95 4D 62 5F 11 CD F0 AC D5 EB 1E DB 0C 3F C4 9E F2 41
3B 30 09 06 07 2A 86 48 CE 3D 04 01 03 38 00 30 35 02 18 08 6E 3D 74 FE 8C 59 1C 0D 7F A7 41 5F 99 DB 5F 37 68 7C C7 AA 72
99 A7 02 19 00 CC E8 C8 BB 09 03 39 3C 60 81 EF 5A 28 4A 40 B7 75 BB F0 1D 10 01 A8 40 11 07 CC CC CC CC CC 10 07 89 EE F9
52 0C 01 0E 47 43 30 31 32 33 34 35 36 37 38 39 03 18 DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD
DD 09 39 30 35 02 18 45 F4 3D AF 99 66 EE 12 04 36 A7 06 3E FE 3A 52 C4 D6 5F 4A CD 0D B5 67 02 19 00 CA AB FB 86 4C DE B7
F8 8D 22 C2 B9 4F A0 58 9F 26 6B 94 06 E2 D6 09 5B
Recv: 69 84
```

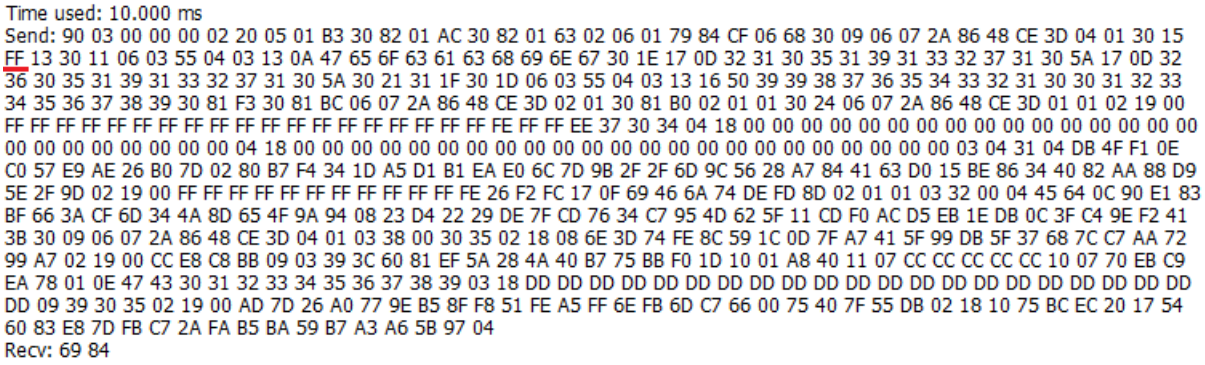
Obr. 2.10: Výpis komunikácie z terminálu - preposlanie chybnej hodnoty náhodného čísla karty.

Test zaslania certifikátu s chybou

Prevedenie testu: Testovacia aplikácia je upravená tak, aby po prijatí náhodného čísla a identifikátora karty odoslala chybný certifikát spolu s korektnou správou a jej podpisom. Chyba v certifikáte sa dosiahne pozmenením jedného bajtu dát certifikátu.
Výsledok: Po výbere appletu (korektnom, keďže odpoveď bola `9000`) terminál korektne zaslal APDU príkaz pre začatie kryptografického protokolu. Ako odpoveď karta odoslala náhodné číslo a svoj identifikátor. Do ďalšieho APDU príkazu, ktorý karta obratom prijala bola vložená chyba do certifikátu hráča zámerne vložená chyba zvýraznená na obr. 2.11). Ako je ďalej z výpisu vidieť, odpoveďou je návratový kód `6984`, ktorý vraví o tom, že údaje v dátovom poli sú neplatné. Chybový návratový kód bol zaslaný po tom, čo výsledok overenia autentickosti certifikátu bol negatívny. Ak by sa chyba vniesla nie do dát certifikátu, ale do podpisu certifikátu, výsledok by bol totožný.

Test zaslania príkazu pre ukončenie behu kryptografického protokolu

Prevedenie testu: Testovacia aplikácia je upravená tak, aby po prijatí náhodného



Obr. 2.11: Výpis komunikácie z terminálu - zaslanie certifikátu s chybou.

čísla a identifikátora karty odoslala karte príkaz na ukončenie kryptografického protokolu. Simuluje tak situáciu, kedy smartfón vyhodnotí, že danú skrýš už hráč v minulosti našiel.

Výsledok: Po výbere appletu (korektnom, keďže odpoveď bola `9000`) terminál korektne zaslal APDU príkaz pre ukončenie kryptografického protokolu. Karta vrátila APDU odpoveď zobrazenú na obr. 2.12. Návratový kód hovorí o úspešnom vykonaní príkazu a teda kryptografický protokol bol ukončený. Ako je ďalej vidieť, bol zaslaný ďalší príkaz pre zahájenie kryptografického protokolu. Nakoľko je už ale táto možnosť blokovaná, karta vrátila chybový návratový kód `6984`.

```
Send: 90 01 00 00
Recv: 10 07 70 EB C9 EA 78 01 0E 47 43 30 31 32 33 34 35 36 37 38 39 90 00
Time used: 10.000 ms
Send: 90 02 00 00
Recv: 90 00
Time used: 10.000 ms
Send: 90 01 00 00
Recv: 69 86
Time used: 10.000 ms
```

Obr. 2.12: Výpis komunikácie z terminálu - príkazu pre ukončenie kryptografického protokolu.

Test zaslania správnych dát

Prevedenie testu: Testovacia aplikácia je upravená tak, aby po prijatí náhodného čísla a identifikátora karty odoslala všetky dáta korektne.

Výsledok: Po výbere appletu (korektnom, keďže odpoveď bola `9000`) terminál korektne zaslal APDU príkaz pre začatie kryptografického protokolu. Ako odpoveď karta odoslala náhodné číslo a svoj identifikátor. Na kartu bolo zaslané APDU obsahujúce certifikát hráča, správu obsahujúcu korektné hodnoty a podpis tejto správy.

Karta obratom odoslala svoj certifikát, správu a jej podpis. Po dôkladnej analýze je na obr. 2.13 možné vidieť, že dáta, ktoré odoslala karta sú korektné. Odpoveď obsahuje taktiež návratový kód 9000, takže spracovanie dát na karte a odoslanie prebehlo v poriadku. Ďalej je vidieť, že na kartu bol zaslaný príkaz pre ukončenie krypografického protokolu, pričom karta opäť zaslala návratový kód 9000.

```
Send: 00 A4 04 00 08 AB CD EF FE DC 12 34 56 00
Recv: 90 00
Time used: 10.000 ms
Send: 90 01 00 00
Recv: 10 07 70 EB C9 EA 78 01 0E 47 43 30 31 32 33 34 35 36 37 38 39 90 00
Time used: 10.000 ms
Send: 90 03 00 00 00 02 20 05 01 B3 30 82 01 AC 30 82 01 63 02 06 01 79 84 CF 06 68 30 09 06 07 2A 86 48 CE 3D 04 01 30 15
31 13 30 11 06 03 55 04 03 13 0A 47 65 6F 63 61 63 68 69 6E 67 30 1E 17 0D 32 31 30 35 31 39 31 33 32 37 31 30 5A 17 0D 32
36 30 35 31 39 31 33 32 37 31 30 5A 30 21 31 1F 30 1D 06 03 55 04 03 13 16 50 39 39 38 37 36 35 34 33 32 31 30 30 31 32 33
34 35 36 37 38 39 30 81 F3 30 81 BC 06 07 2A 86 48 CE 3D 02 01 30 81 B0 02 01 01 30 24 06 07 2A 86 48 CE 3D 01 01 02 19 00
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FE FF FF EE 37 30 34 04 18 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 04 18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 04 31 04 DB 4F F1 0E
C0 57 E9 AE 26 B0 7D 02 80 B7 F4 34 1D A5 D1 B1 EA E0 6C 7D 9B 2F 2F 6D 9C 56 28 A7 84 41 63 D0 15 BE 86 34 40 82 AA 88 D9
5E 2F 9D 02 19 00 FF FF FF FF FF FF FF FF FF FF FF FE 26 F2 FC 17 0F 69 46 6A 74 DE FD 8D 02 01 01 03 32 00 04 45 64 0C 90 E1 83
BF 66 3A CF 6D 34 4A 8D 65 4F 9A 94 08 23 D4 22 29 DE 7F CD 76 34 C7 95 4D 62 5F 11 CD F0 AC D5 EB 1E DB 0C 3F C4 9E F2 41
3B 30 09 06 07 2A 86 48 CE 3D 04 01 03 38 00 30 35 02 18 08 6E 3D 74 FE 8C 59 1C 0D 7F A7 41 5F 99 DB 5F 37 68 7C C7 AA 72
99 A7 02 19 00 CC E8 C8 BB 09 03 39 3C 60 81 EF 5A 28 4A 40 B7 75 BB F0 1D 10 01 A8 40 11 07 CC CC CC CC CC 10 07 70 EB C9
EA 78 01 0E 47 43 30 31 32 33 34 35 36 37 38 39 03 18 DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD
DD 09 39 30 35 02 19 00 AD 7D 26 A0 77 9E B5 8F F8 51 FE A5 FF 6E FB 6D C7 66 00 75 40 7F 55 DB 02 18 10 75 BC EC 20 17 54
60 83 E8 7D FB C7 2A FA B5 BA 59 B7 A3 A6 5B 97 04
Recv: 05 01 AA 30 82 01 A3 30 82 01 59 02 06 01 79 84 C6 57 AB 30 09 06 07 2A 86 48 CE 3D 04 01 30 15 31 13 30 11 06 03 55
04 03 13 0A 47 65 6F 63 61 63 68 69 6E 67 30 1E 17 0D 32 31 30 35 31 39 31 33 31 37 34 31 5A 17 0D 32 36 30 35 31 39 31 33
31 37 34 31 5A 30 17 31 15 30 13 06 03 55 04 03 13 0C 47 43 30 31 32 33 34 35 36 37 38 39 30 81 F3 30 81 BC 06 07 2A 86 48
CE 3D 02 01 30 81 B0 02 01 01 30 24 06 07 2A 86 48 CE 3D 01 01 02 19 00 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FE FF FF EE 37 30 34 04 18 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 04 18 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 04 31 04 DB 4F F1 0E C0 57 E9 AE 26 B0 7D 02 80 B7 F4 34 1D A5 D1 B1 EA
E0 6C 7D 9B 2F 2F 6D 9C 56 28 A7 84 41 63 D0 15 BE 86 34 40 82 AA 88 D9 5E 2F 9D 02 19 00 FF FF FF FF FF FF FF FF FF FF FE
26 F2 FC 17 0F 69 46 6A 74 DE FD 8D 02 01 01 03 32 00 04 C8 E9 3C FA 55 46 D2 E8 94 F2 2E F5 61 3D 83 AF 8A 1B 7F 90 EF 2F
08 DD 74 C4 3D 08 7D 16 EF 72 C9 6A FF F9 52 3C F6 F3 78 52 86 C5 11 E8 23 88 30 09 06 07 2A 86 48 CE 3D 04 01 03 39 00 30
36 02 19 00 D0 3E 56 6C F7 2B 45 90 24 46 AC C4 60 21 F5 C0 CB 11 07 73 7C 02 E4 62 02 19 00 90 AC 94 85 54 4B 36 68 A7 DA
5B BE F7 B4 DB 3F 32 5D 68 8D 7A D9 A8 F1 10 07 70 EB C9 EA 78 11 07 CC CC CC CC CC 01 0E 47 43 30 31 32 33 34 35 36 37 38
39 02 18 50 39 39 38 37 36 35 34 33 32 31 30 30 31 32 33 34 35 36 37 38 39 14 04 00 01 04 18 54 47 34 34 34 34 34 34 34 34
34 34 34 34 34 34 34 34 34 34 34 03 18 DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD DD 08 3A 30
36 02 19 00 A8 20 17 76 D0 F9 EA 28 18 D2 FB 6E FE 50 BC BA E7 13 86 60 05 A0 1F A4 02 19 00 C6 45 0C 15 1D 82 68 13 98 7C
21 29 88 1B 46 3A C5 81 CD 3E D5 D1 AB AE 90 00
Time used: 100.000 ms
Send: 90 02 00 00
Recv: 90 00
Time used: 10.000 ms
```

Obr. 2.13: Výpis komunikácie z terminálu - zaslanie správnych dát pre beh kryptografického protokolu.

2.2.2 Test aplikácie pre smartfón

Pri testovaní tejto aplikácie bolo skontrolované, či smartfón dokáže korektne karte zasielať príkazy a taktiež od nej prijímať odpovede. Tento test sa dá rozdeliť do niekoľkých krokov na základe návrhu komunikácie.

Test nadviazania spojenia smartfónu s kartou

Prevedenie testu: Pre nadviazanie spojenia je potrebné priblížiť smartfón ku karte.

Výsledok: Bezprostredne po tom, čo smartfón prišiel do kontaktu s kartou, sa indikátor pripojenia karty zmenil a informoval o tom, že smartfón je v spojení so skrýšou (viď. obr. 2.14). K tomu došlo na základe toho, že na karte nastal reset a odoslala smartfónu ATR.

Obr. 2.14: Zmena indikátora pripojenia karty pred a po kontakte s kartou.

Test zaznamenania skrýše

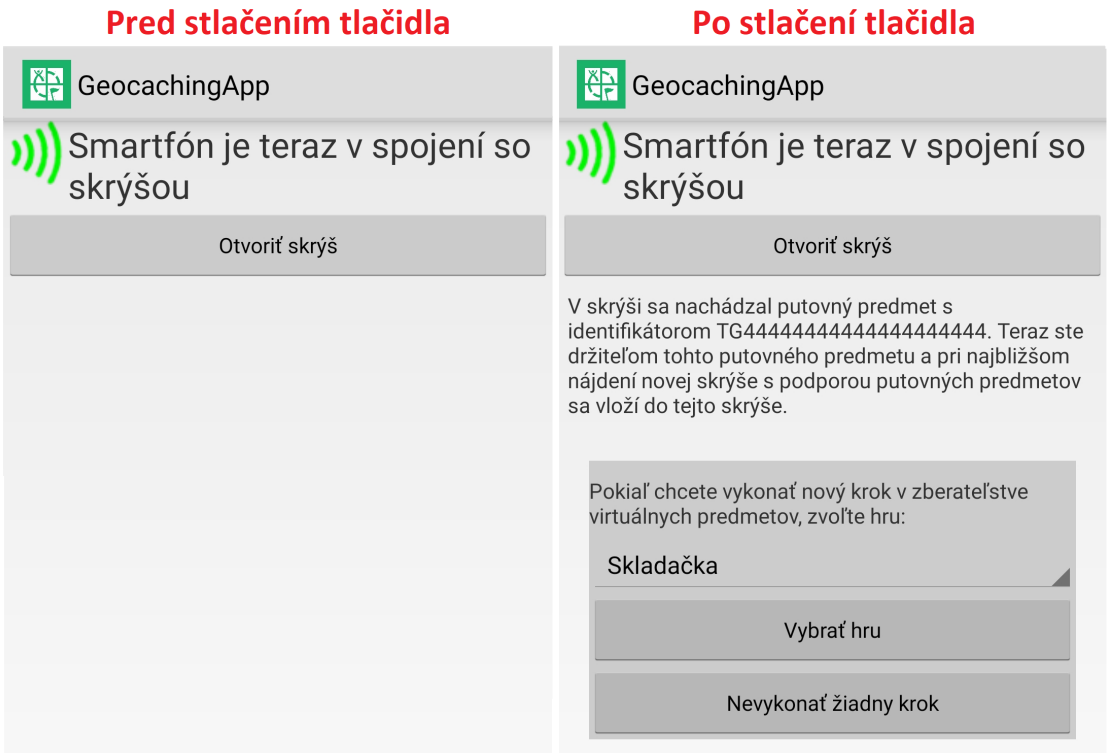
Prevedenie testu: Pre zaznamenanie skrýše, čo predstavuje nadviazanie spojenia a zbehnutie kryptografického protokolu, je potrebné byť v spojení s kartou a stlačiť tlačidlo „Otvoriť skrýš".
Výsledok: Po stlačení tlačidla sa automaticky previedli všetky kroky kryptografického protokolu a korektne sa zobrazilo vyskakovacie okno pre možnosť previesť krok v zberateľstve virtuálnych predmetov. Taktiež bola zobrazená správa o predaní putovného predmetu (viď. obr. 2.15.

Test zaznamenania skrýše a prevedenie kroku v skladačke

Prevedenie testu: Po tom, čo úspešne prebehne zaznamenanie skrýše sa zo zoznamu zvolí možnosť hry s názvom „Skladačka".
Výsledok: Po stlačení tlačidla sa okno automaticky zatvorí. Zobrazí sa správa o prevedení kroku v danej hre a zobrazí sa získaný obrázok (viď. obr. 2.16).

Test zaznamenania skrýše a prevedenie kroku v zberateľstve predmetov

Prevedenie testu: Po tom, čo úspešne prebehne zaznamenanie skrýše sa zo zoznamu zvolí možnosť hry s názvom „Prírodné javy".



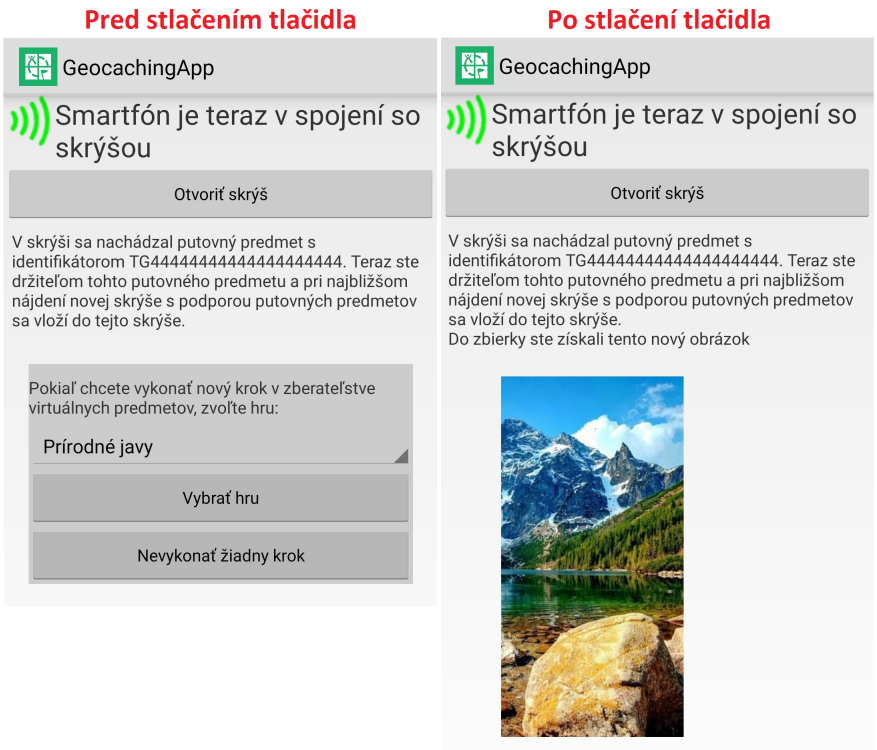
Obr. 2.15: Zobrazenie vyskakovacieho okna pre prevedenie kroku v zberateľstve virtuálnych predmetov.

Výsledok: Po stlačení tlačidla sa okno automaticky zatvorí. Zobrazí sa správa o prevedení kroku v danej hre a zobrazí sa získaný obrázok (viď. obr. 2.17).

Test zaznamenania skrýše a neprevedenia žiadnej hry

Prevedenie testu: Po tom, čo úspešne prebehne zaznamenanie skrýše sa klikne na možnosť „Nevykonať žiadny krok“.
Výsledok: Po stlačení tlačidla sa okno automaticky zatvorí. Ostane zobrazená iba správa o predaní putovného predmetu (viď. obr. 2.18.

Obr. 2.16: Prevedenie kroku v hre „Skladačka“.



Obr. 2.17: Prevedenie kroku v hre „Prírodné javy“.

Obr. 2.18: Neprevedenie kroku v žiadnej hre.

Záver

V rámci tejto práce bolo úlohou vytvoriť aplikácie pre bezkontaktnú čipovú kartu a smartfón, ktoré medzi sebou budú komunikovať pomocou rozhrania NFC. Komunikácia má prebiehať pomocou kryptografického protokolu navrhnutého na základe priloženej špecifikácie zadania. Táto komunikácia je prispôsobená k použitiu v rámci geokešingu, kde bezkontaktná karta predstavuje skrýšu. K tomu bolo potrebné naštudovať a popísať základy programovania mikropočítačových bezkontaktných kariet a problematiku použitia eliptických kriviek v kryptografii. Daná téma bola preštudovaná a na základe získaných poznatkov bol navrhnutý model komunikácie medzi týmito dvoma zariadeniami. Na základe tohto modelu bol vytvorený návrh oboch aplikácií a pomocou neho vytvorené aplikácie.

Funkčnosť oboch aplikácií bola otestovaná. Výsledky testov boli zhrnuté. Zo získaných výsledkov je zjavné, že aplikácie fungujú a dokážu reálne predviesť požadovanú zabezpečenú komunikáciu medzi smartfónom a bezkontaktnou kartou.

Súčasťou tejto práce nebolo vytvoriť herný server a teda aj napriek tomu, že sú aplikácie funkčné, nebolo by zatiaľ možné ich uviesť do reálnej prevádzky. Poskytujú ale základ, ktorý môže byť použitý na ďalší vývoj a v konečnom dôsledku predstavujú základ pre plnohodnotné aplikácie, ktoré môžu byť použité v rámci geokešingu, kde bude fyzickú skrýš nahrádzať práve bezkontaktná čipová karta.

Literatúra

[1] BURDA K.: *Bezkontaktní mikropočítačová karta jako skrýš pro geokešing*. Elektrorevue, Brno, 2020.

[2] Wiki Geocaching. *Historie* [online]. [cit. 2021-3-14]. Dostupné z: http://wiki.geocaching.cz/wiki/Historie

[3] Wiki Geocaching. *Jak začít* [online]. [cit. 2021-3-14]. Dostupné z: http://wiki.geocaching.cz/wiki/Jak_začít

[4] Wiki Geocaching. *Geocache* [online]. [cit. 2021-3-14]. Dostupné z: http://wiki.geocaching.cz/wiki/Cache

[5] RANKL W., EFFING W.: *Smart card handbook*. Chichester: Wiley, 2003. ISBN 0-470-85668-8

[6] DICHOU K., TOURTCHINE V., RAHMOUNE F.: Simulation of APDUs exchanged between a microcontroller smart card and a reader. In: *2015 7th International Conference on Modelling, Identification and Control (ICMIC)* [online]. IEEE, 2015, 2015, s. 1-4 [cit. 2020-12-11]. ISBN 978-0-9567-1575-3. Dostupné z: doi:10.1109/ICMIC.2015.7409426

[7] HANSMANN U., NICKLOUS M., SCHÄCK T., SCHNEIDER A., Seliger F.: *Smart card application development using Java*. 2nd ed. Berlin: Springer, 2002. ISBN 978-3-540-43202-9.

[8] ROLAND M.: *Security Issues in Mobile NFC Devices*. Cham: Springer International Publishing, 2015 [cit. 2020-12-10]. T-Labs Series in Telecommunication Services. ISBN 978-3-319-15487-9. Dostupné z: doi:10.1007/978-3-319-15488-6

[9] NFC Forum. *Device Requirements* [online]. 2017-02-20 C[cit. 2020-12-05]. Dostupné z: https://nfc-forum.org/wp-content/uploads/2017/08/NFCForum-DeviceRequirements-1.5.02.pdf

[10] AKRAM R. N., MARKANTONAKIS K., MAYES K.: *An Introduction to Java Card Programming*. MARKANTONAKIS, Konstantinos a Keith MAYES, ed. Secure Smart Embedded Devices, Platforms and Applications [online]. New York, NY: Springer New York, 2014, 2014-9-13, s. 497-513 [cit. 2020-12-05]. ISBN 978-1-4614-7914-7. Dostupné z: doi:10.1007/978-1-4614-7915-4_22

[11] Developing NFC Applications with Android. LESAS A., MIRANDA S.: *The Art and Science of NFC Programming*. Hoboken, NJ, USA: John Wiley & Sons,

2016, 2017-01-06, s. 45-105 [cit. 2020-12-10]. ISBN 9781119379072. Dostupné z: doi:10.1002/9781119379072.ch2

[12] Android Developers. *Activity* [online]. [cit. 2020-12-05]. Dostupné z: https://developer.android.com/reference/android/app/Activity

[13] Android Developers. *Intent* [online]. [cit. 2020-12-05]. Dostupné z: https://developer.android.com/reference/android/content/Intent

[14] Android Developers. *IsoDep* [online]. [cit. 2020-12-05]. Dostupné z: https://developer.android.com/reference/android/nfc/tech/IsoDep

[15] *SEC 2: Recommended Elliptic Curve Domain Parameters* [online]. 2010-01-27, [cit. 2021-5-20]. Dostupné z: https://www.secg.org/sec2-v2.pdf

[16] PORNIN T.: *RFC 6979 - Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)* [online]. 2013-08, [cit. 2021-5-20]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc6979

[17] NAKOV S.: *Practical Cryptography for Developers: ECDSA: Elliptic Curve Signatures* [online]. Sofia, 2018 [cit. 2021-5-20]. Dostupné z: https://cryptobook.nakov.com/digital-signatures/ecdsa-sign-verify-messages

[18] KENT S.: *RFC 1422 - Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management* [online]. 1993-02, [cit. 2021-5-20]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc1422

[19] BURDA K.: *Bezpečnost informačních systému*. Prvé vydanie. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, Technická 12, 616 00 Brno, 2013. ISBN 978-80-214-4890-2.

[20] ESTAKLE D., JONES P.: *RFC 3174 -US Secure Hash Algorithm 1 (SHA1)* [online]. 2001-09, [cit. 2021-5-20]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc3174

Zoznam symbolov, veličín a skratiek

ACK potvrdenie – Acknowledgement

AID identifikátor aplikácie – Application ID

APDU dátová jednotka aplikačného protokolu – Application Protocol Data Unit

API rozhranie pre programovanie aplikácií – Application Programming Interface

ATR odpoveď na reset požiadavku – Answer To Reset

AVP dvojica atribút-hodnota – Attribute-Value Pair

BGT ochranný čas bloku – Block Guard Time

BWT čas čakania na blok – Block Waiting Time

CA certifikačná autorita

C-APDU príkazová dátová jednotka aplikačného protokolu – Command Application Protocol Data Unit

CLA trieda – Class

CRC kontrola cyklickej redundancie – Cyclic Redundancy Check

CWT čas čakania na znak – Character Waiting Time

DER Rozlišujúce pravidlá kódovania – Distinguished Encoding Rules

DSA Protokol digitálneho podpisu – Digital Signature Algorithm

ECDSA Protokol digitálneho podpisu s využitím eliptických kriviek – Elliptic Curve Digital Signature Algorithm

EDC kód detekcie chyby – Error Detection Code

EEPROM elektricky mazateľná permanentná pamäť – Electrically Erasable Programable Read-Only Memory

ETU elementárna časová jednotka – Elementary Time Unit

GPS globálny lokalizačný systém – Global Positioning System

H Hráč – Player

IDE	integrované vývojové prostredie – Integrated Development Environment
INS	inštrukcia – Instruction
JCRE	behové prostredie Java Card – Java Card Runtime Environment
Lc	dlžka dát v príkaze – Length command
Le	dlžka dát v očakávanej odpovedi – Length expected
LEN	dlžka – Length
LLCP	Protokol kontroly logického spojenia – Logical Link Control Protocol
LRC	kontrola pozdlžnej redundancie – Longitudinal Redundancy Check
M bit	bit informujúci o viac dátach – More Data bit
N	Poradie hráča
NAD	adresa uzla – Node Address
NFC	Near Field Communication
NIST	Národný inštitút štandardov a technológie – National Institute of Standards and Technology
PCB	riadiaci bajt protokolu – Protocol Control Byte
PPS	výber parametrov protokolu – Protocol Parameter Selection
R-APDU	odpoveďová dátová jednotka aplikačného protokolu – Response Application Protocol Data Unit
RAM	pamäť s priamym prístupom – Random Access Memory
RFID	Rádiofrekvenčná identifikácia – Radio Frequency Identification
R_H	náhodné číslo hráča
RM OSI	referenčný model prepojenia otvorených systémov – Open System Interconnection Reference Model
ROM	permanentná pamäť – Read-Only Memory
R_s	Náhodné číslo skrýše – Cache Random Number
S	Skrýša – Cache

SHA-1	Zabezpečený hešovací algoritmus 1 – Secure Hash Algorithm 1
SK	súkromný kľúč
SW	návratový kód – Status Word
VK	verejný kľúč
VK_A	verejný kľúč účastníka A
VK_{CA}	verejný kľúč certifikačnej autority

Zoznam príloh

A Špecifikácia pre diplomovú prácu

Specifikace pro diplomovou práci Bc. Damiána Vertaľa

Doc. K. Burda, CSc.

24.1.2021

1. Úvodní poznámky

Ke zvýšení atraktivity geokešingu se předpokládá možnost organizovat dva typy her. Prvním typem je putování virtuálních předmětů a druhým typem je sběratelství virtuálních předmětů. V případě putování virtuálních předmětů má každý hráč H možnost si vytvořit a u provozovatele herního serveru HS zaregistrovat svůj putovní virtuální předmět (například obrázek) s identifikátorem *U*. Putovní předmět se vyznačuje tím, že při každém nálezu skrýše se automaticky předává mezi smartfonem nálezce a skrýší. Ze smartfonu hráče H je tedy jeho putovní předmět předán do skrýše S, při nálezu této skrýše dalším hráčem X je tento putovní předmět předán do smartfonu hráče X atd. Putovní předmět tak postupně přes smartfony jednotlivých hráčů putuje různými skrýšemi po celém světě, což hráči mohou sledovat prostřednictvím herního serveru.

Druhým typem hry je sběratelství virtuálních předmětů. Provozovatel herního serveru definuje různé hry a hráč si ve své aplikaci zvolí, kterou z nich chce právě hrát. S každým nálezem skrýše pak hráčova aplikace vykoná jeden krok hry. Pokud se hráč rozhodl hrát skládačku, tak s každým nálezem nové skrýše mu herní aplikace zpřístupní nový dílek skládačky. Pokud hráč shromažďuje sérii nějakých virtuálních předmětů (např. obrázky historických aut), tak mu herní aplikace při nálezu skrýše zpřístupní předmět, který ve své sbírce ještě nemá.

2. Kryptografický protokol

K přenosu jednotlivých hodnot v níže uvedeném protokolu se požaduje použít formát AVP (Attribute–Value Pair). Každá přenášená hodnota (např. identita H, certifikát *CRT*, náhodná hodnota *R* atd.) se přenáší jako trojice $Y = (A, L, V)$. První bajt A reprezentuje typ hodnoty (např. Identita) a druhý bajt L udává celkovou délku trojice Y v bajtech. Zbytek $(L - 2)$ bajtů je samotná hodnota V (například textový řetězec dané identity). Číslování typů je zapotřebí volit, aby bylo systematické. Inspiraci k tomu lze nalézt např. v RFC 2865.

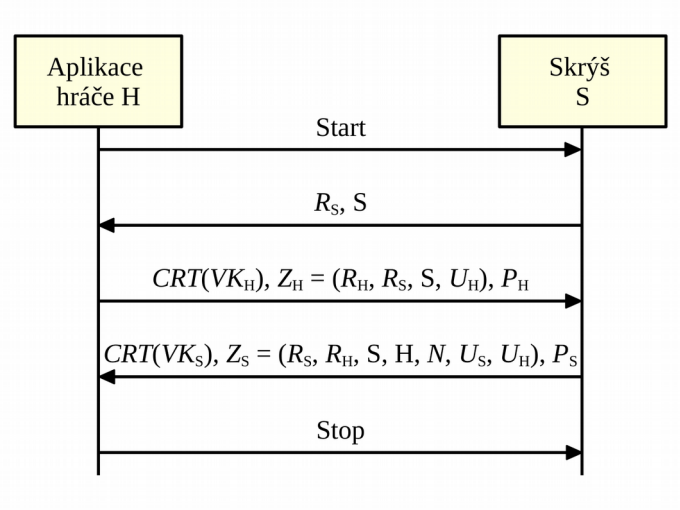
K podpisování zpráv požaduje použít techniku ECDSA 192 bitů a délka náhodných čísel musí být alespoň 32 bitů. Skrýš i hráčův smartfon mají v sobě bezpečně uložen veřejný klíč VK_{CA} certifikační autority.

Identifikátory jsou unikátním textovým řetězcem v kódování ASCII. Identifikátor skrýše je o délce nejvýše 12 bajtů, přičemž první dva bajty jsou znaky GC. Identifikátor putovního předmětu a hráče jsou řetězce o délce nejvýše 30 bajtů, přičemž první dva bajty identifikátoru putovního předmětu jsou znaky TG. Hráčův smartfon si vede seznam, do něhož ukládá identifikátory již nalezených skrýší, tj. skrýší, které mu vydaly potvrzení o jejich nálezu.

Nyní k samotnému běhu protokolu (viz obr. 1). Po ustavení přenosového kanálu mezi smartfonem a skrýší odešle hráčova aplikace zprávu Start, čímž zahájí běh protokolu. Skrýš pak protistraně odešle náhodné číslo R_S a svůj identifikátor S. Hráčova aplikace nejprve ve svém seznamu zkontroluje podle identifikátoru S, zda tato skrýš již nebyla hráčem nalezena. V kladném případě hráčova aplikace běh protokolu ukončí příkazem Stop a v opačném případě se v protokolu pokračuje. Uvedené opatření slouží k tomu, aby skrýš vydala každému nálezci jen jediné potvrzení o svém nálezu.

Smartfon poté skrýši odešle hráčův certifikát $CRT(VK_H)$ a dvojici Z_H, P_H, kde Z_H je hráčova zpráva a P_H je hráčův podpis této zprávy. Skrýš si nejprve pomocí veřejného klíče VK_{CA} certifikační autority ověří platnost hráčova certifikátu. Pokud je vše v pořádku, tak hráčův veřejný klíč VK_H z tohoto

certifikátu použije k ověření, zda podpis P_H je hráčův podpis zprávy Z_H. Jestliže je zmiňovaný podpis autentický, tak tím si skrýš ověřila identitu hráče H (jeho identifikátor zná z certifikátu) i autentičnost přijaté zprávy $Z_H = (R_H, R_S, S, U_H)$. V uvedené zprávě si skrýš ověří správnost jak hodnoty náhodného čísla R_S, tak i svého identifikátoru S a také si zapíše hodnotu R_H, což je náhodné číslo, které pro daný běh protokolu vygenerovala hráčova aplikace. Čísla R_H a R_S jsou vzhledem ke své náhodnosti pro daný běh protokolu unikátní a kontrola jejich hodnot znemožňuje útoky využitím dat, která byla přenesena v jiných běžích protokolu. Hodnota U_H je identifikátorem putovního předmětu, uloženém ve smartfonu hráče. Pokud je tato hodnota nula, tak to znamená, že se ve smartfonu hráče žádný putovní předmět nenachází.



Obr. 1: Kryptografický protokol pro komunikaci mezi aplikací hráče a skrýší

Skrýš poté vygeneruje potvrzení o jejím nálezu, což je zpráva $Z_S = (R_S, R_H, S, H, N, U_S, U_H)$, pro níž pomocí svého soukromého klíče vytvoří podpis P_S. Hodnota N je přitom pořadové číslo hráče H v postupnosti všech hráčů, kteří skrýš doposud objevili a hodnota U_S je identifikátorem předmětu, který je ve skrýši uložen. Pokud je tato hodnota nula, tak se ve skrýši žádný putovní předmět nenachází.

Následně skrýš odešle protistraně trojici $CRT(VK_S)$, Z_S, P_S. Hráčova aplikace nejprve pomocí veřejného klíče VK_{CA} certifikační autority ověří platnost přijatého certifikátu skrýše. Pokud je vše v pořádku, tak se z tohoto certifikátu použije veřejný klíč VK_S k ověření, zda podpis P_S je podpisem skrýše S pro zprávu Z_S. Je-li podpis autentický, tak tím si hráčova aplikace ověří identitu skrýše S i autentičnost zprávy Z_S.

Aplikace poté zahájí zpracování zprávy $Z_S = (R_S, R_H, S, H, N, U_S, U_H)$. V této zprávě nejprve ověří správnost hodnot obou náhodných čísel R_S a R_H a rovněž tak správnost identit S a H. Zpráva je prakticky potvrzením skrýše S, že hráč H je N–tým nálezcem této skrýše a v rámci tohoto nálezu došlo k výměně putovních předmětů tak, že ve skrýši je nyní uložen předmět s kódem U_H a ve smartfonu hráče se nyní nachází předmět s kódem U_S. Nakonec hráčova aplikace zašle skrýši příkaz Stop, čímž běh protokolu ukončí. Po přijetí tohoto příkazu skrýš inkrementuje hodnotu N a uloží si ji pro následujícího nálezce. Příkazem Stop může ukončit běh protokolu i skrýš. Tato situace nastane, když je negativní některá z kontrol dat přijatých z hráčovy aplikace. V takovémto případě je zapotřebí hráčův smartfon od skrýše nejprve oddálit a poté k ní opět přiblížit. Tím se spustí nový běh protokolu.

Hráčova aplikace ověří podpis P_S zprávy Z_S a správnost hodnot v této zprávě. Pokud je vše v pořádku, tak stanoveným způsobem aktualizuje stav hráčovy hry.

Poté se ještě uskuteční následující kroky, které však již nejsou součástí zadání práce. Hráčova aplikace přepošle přijatou trojici $CRT(VK_S)$, Z_S, P_S hernímu serveru HS. Herní server provede kontrolu zaslaného certifikátu i podpisu. Je-li vše v pořádku, tak hráče H zveřejní jako N–tého nálezce skrýše S. A pokud je hodnota U_S, resp. U_H nenulová, tak rovněž zveřejní, že hráč H nyní disponuje putovním předmětem U_S, resp., že ve skrýši S je nyní uložen putovní předmět U_H.

B Android aplikácia

Priložený .zip súbor obsahuje inštalačný APK súbor naprogramovanej Android aplikácie.

C Zdrojové kódy Android aplikácie

Priložený .zip súbor obsahuje zdrojové kódy pre Android aplikáciu GeocachingApp.

D Zdrojový kód appletu pre Java Card

Priložený .zip súbor obsahuje zdrojový kód appletu pre čipovú kartu vo formáte .java.