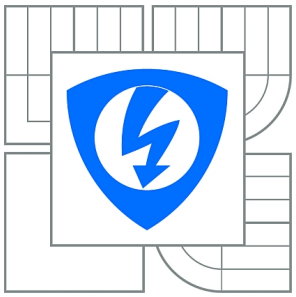
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

AUTENTIZACE POMOCÍ ZAŘÍZENÍ S OS ANDROID

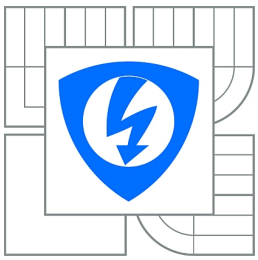
AUTHENTICATION USING A DEVICE WITH OS ANDROID

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MARTIN SMÍŠEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. PETR DZURENDA
SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Martin Smíšek	***ID:***	153163
Ročník:	3	***Akademický rok:***	2014/2015

NÁZEV TÉMATU:

Autentizace pomocí zařízení s OS Android

POKYNY PRO VYPRACOVÁNÍ:

Student se zaměří na možnosti využití zařízení s operačním systémem Android při autentizaci osob v přístupových systémech. Důraz bude kladen na technologie NFC (Near Field Communication) a BLE (Bluetooth Low Energy). Student se také zaměří na možnosti využití tzv. secure elementu pro bezpečné uložení dat v těchto zařízeních. Výstupem práce pak bude návrh a implementace vlastního autentizačního systému využívajícího tyto technologie, včetně zhodnocení bezpečnosti navrženého systému.

DOPORUČENÁ LITERATURA:

[1] MURPHY, Mark L. Android 2: Průvodce programováním mobilních aplikací. Vyd. 1. Brno: Computer Press, 2011, 375 s. ISBN 978-80-251-3194-7.

[2] MENEZES, Alfred J, Paul C OORSCHOT a Scott A VANSTONE. Handbook of applied cryptography. Vyd. 1. Boca Raton: CRC Press, 1997, 780 s. ISBN 0-8493-8523-7.

[3] STALLINGS, William. Cryptography and network security: principles and practice. Seventh edition. xix, 731 pages. ISBN 01-333-5469-5.

Termín zadání:	9.2.2015	***Termín odevzdání:***	2.6.2015

Vedoucí práce: Ing. Petr Dzurenda
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Abstrakt

Bakalářská práce popisuje základy kryptografie zahrnující symetrickou a asymetrickou kryptografii. Dále popisuje metody autentizace znalostí, žadatelem a předmětem. Zabývá se popisem perspektivních technologií NFC a BLE a naznačuje, jak v OS Android vyvíjet aplikace, které pracují s těmito technologiemi. Uvádí, jak je možné bezpečně uložit klíče a hesla v rámci OS Android. V práci je navržen a implementován vlastní přístupový systém, který využívá k autentizaci zařízení s OS Android a technologii NFC. Systém je rozdělen na dvě části. První část pracující na zařízení s OS Android, která generuje, resetuje a bezpečně ukládá tajná data a zajišťuje přenos dat. A druhou část pracující v terminálu s připojenou RFID/NFC čtečkou. Jejím úkolem je komunikace se zařízením přiloženým k čtečce a zajištění centrálního uložení dat o uživatelích a místnostech na webovém serveru. Zajišťuje také vytváření logů ve formě textových souborů, které jsou uloženy na FTP server.

Klíčová slova

Android, Autentizace, BLE, NFC, Kryptografie, Přístupový systém, Řízení přístupu

Abstract

This thesis describes the basics of cryptography including symmetric and asymmetric cryptography. It also looks into methods of authentication, using knowledge of some information, having any object and user specific proportions. The thesis engage in description of modern technologies NFC and BLE and shows, how to develop applications, working with them. It deals with the way of storing data used for cryptography, in the safe way. The access system created in the scope of this thesis, which is using Android powered device as an authentication object and communicating over NFC is described. The system can be divided into two parts. The first part works on the device powered by Android. It creates, resets and saves secret data safely and communicate with the other part. The second part is working in a terminal with connected RFID/NFC card reader. The task of this part is to communicate with the part on the Android device, attached to the reader and saving data of users and chambers centrally on the web server. It also creates logs in the form of text files and upload it to the FTP server.

Keywords

Access management, Access system, Android, Authentication, BLE, Cryptography, NFC

SMÍŠEK, M. *Autentizace pomocí zařízení s OS Android*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2015. 67 s. Vedoucí bakalářské práce Ing. Petr Dzurenda.

Prohlášení

Prohlašuji, že svoji bakalářskou práci na téma Autentizace pomocí zařízení s OS Android jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením tohoto projektu jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009Sb.

V Brně dne ....

Martin Smíšek

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Petru Dzurendovi za velmi přínosnou pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce. Dále děkuji rodičům, kteří mě podporovali po celou dobu mého studia.

V Brně dne ....

Martin Smíšek

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication

Brno University of Technology
Technicka 12, CZ-61600 Brno, Czechia

http://www.six.feec.vutbr.cz

Výzkum popsaný v této bakalářské práci byl realizovaný v laboratořích podpořených projektem Centrum senzorických, informačních a komunikačních systémů (SIX); registrační číslo CZ.1.05/2.1.00/03.0072, operačního programu Výzkum a vývoj pro inovace.

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

Obsah

Seznam obrázků

Seznam tabulek

Úvod

S rozvojem moderních technologií a možností jejich distribuce přes Internet, kdy data mají velmi vysokou cenu, je třeba je nějakým způsobem zabezpečit a zajistit, že pouze vybraní uživatelé budou moci s daty manipulovat.

Mobilní telefony v poslední době také zaznamenaly znatelný pokrok. Současné mobilní telefony dovolují uživateli komunikovat prostřednictvím sítě Internet, stahovat z něj data, spravovat internetové bankovnictví a tak dále. Umožňují také uchovávat jisté množství dat, která mohou mít pro uživatele (nebo pro případného útočníka) určitou cenu. Jedná se například o kontakty, osobní údaje, přístupová hesla a klíče. Proto je nutné chránit i tato zřízení. Jejich ochrana je ovšem složitá. Útočník může data získat, pokud je telefon vysílá. Musí se také počítat s tím, že telefon lze ukrást. Zároveň je nežádoucí zatěžovat uživatele neustálými autentizačními dotazy (například zadání hesla pro vstup do seznamu kontaktů). Mobilní telefony také stále častěji nacházejí uplatnění v přístupových systémech, kde nahrazují čipové karty. Pokud by se tedy útočník takového telefonu zmocnil, mohl by přistupovat do příslušných oblastí. Tato bakalářská práce se touto problematikou zabývá. V kapitole 1 je představen úvod do kryptografie. V další kapitole je konkrétně popsána autentizace a autentizační metody, rozlišené podle způsobu, jakým uživatelé prokazuji svou identitu. Následně jsou rozebrány a srovnány moderní komunikační technologie, které jsou implementovány do operačního systému Android, konkrétně BLE (*Bluetooth Low Energy*) a NFC (*Near Field Communication*). Praktická část práce se zabývá vlastním návrhem a implementací systému fyzického řízení přístupu, skládající se ze tří aplikací, které byly vytvořeny jako demonstrace autentizace pomocí mobilního telefonu využívající technologii NFC. Tyto aplikace společně demonstrují využití zařízení s OS Android jako autentizačního předmětu. Dvě z nich pracují na zařízení s OS Android, třetí potom slouží k obsluze čtečky karet, správě databáze a FTP (*File Transfer Protocol*) serveru. Aplikace pro OS Android plní stejnou funkci, pracují ovšem v jiným módu (peer to peer a emulace karet). Tajné klíče jsou v zařízení bezpečně uloženy pomocí Android keystore.

1 Kryptografie

Kryptografie je věda, která se zabývá metodami, kterými lze převést data do podoby, ve které budou čitelná pouze se speciální znalostí (dešifrovacím klíčem). Využívá se zde matematických výpočtů, které bez této znalosti nelze se současnými výpočetními prostředky v reálném čase řešit. Je zřejmé, že s rozvojem výpočetních zařízení s vyšším výpočetním výkonem je třeba volit stále složitější výpočty. Toho se většinou dosáhne prodloužením šifrovacích a dešifrovacích klíčů.

Kryptografie pracuje s dvěma druhy klíčů, šifrovacími a dešifrovacími. Odesílatel zprávy musí znát šifrovací klíč. Pomocí toho klíče zprávu zašifruje. Vzniklá zpráva se nazývá kryptogram. Tento kryptogram je odeslán rizikovým kanálem protější straně. Příjemce zprávy potom kryptogram dešifruje pomocí dešifrovacího klíče a získá tak původní zprávu v otevřené podobě. Při průchodu kryptogramu rizikovým kanálem může tento kryptogram však zachytit i případný útočník. Pokud ovšem nezná dešifrovací klíč (a šifra je bezpečná), tak z něj není schopen odvodit původní zprávu.

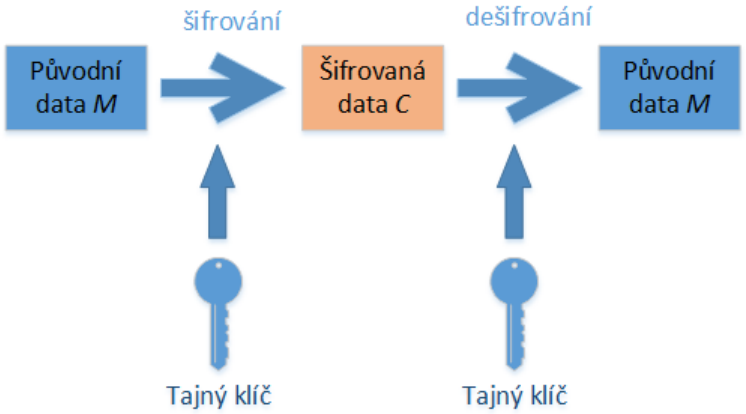
Kryptografie poskytuje dvě základní služby. Zajišťuje autentičnost dat (případně entity) a důvěrnost (utajení) dat. Autentičnost je proces ověření, zda data skutečně pochází od očekávaného odesilatele. Integrita potom zaručuje, že data jsou ve formě, v jaké je odesilatel odeslal. Důvěrnost dat spočívá v tom, že data jsou zašifrována a jsou tedy čitelná pouze pro uživatele vlastnící patřičný dešifrovací klíč. Kryptografii lze dělit na symetrickou a asymetrickou, viz následující kapitoly. V kapitole 1 bylo čerpáno z [6] [8][12][17], matematické základy kryptografie lze nalézt v [15].

Dalšími významnými pojmy v této oblasti jsou **kryptoanalýza**, což je věda, která se snaží získat zabezpečená data ze zachyceného kryptogramu. Snaží se tedy získat obsah zašifrovaných zpráv bez znalosti klíče. **Steganografie** je vědní disciplína, která k utajení komunikace využívá skrytí zpráv. Bezpečnost je tedy založena na tom, že útočník se vůbec o probíhající komunikaci nedozví. Většinou se steganografie kombinuje s kryptografií. Kryptografie, kryptoanalýza a steganografie jsou pak hlavními součástmi vědní disciplíny zvané kryptologie.

1.1 Symetrická kryptografie

V symetrické kryptografii platí, že šifrovací klíč je stejný, jako klíč dešifrovací, nebo je z něj odvoditelný. Odvodit jej lze v reálném čase. Symetrická kryptografie je rychlá a používá kratší klíče než kryptografie asymetrická. V současné době se považuje za bezpečný klíč dlouhý minimálně 112 bitů, v praxi se ovšem využívá většinou klíč délky 128 bitů (s rostoucí délkou bezpečnost roste). Nevýhodou je pak složitá distribuce klíčů. Pokud je třeba

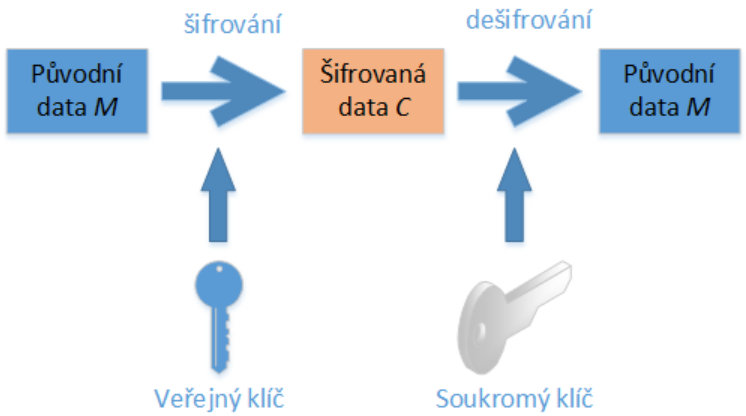
komunikovat s velkým množstvím uživatelů, musí každý z nich znát tajný klíč, který navíc musí být odeslán bezpečným kanálem. Princip symetrické kryptografie je zobrazen na obr. 1.



Obr. 1 Princip šifrování pomocí symetrické kryptografie

1.2 Asymetrická kryptografie

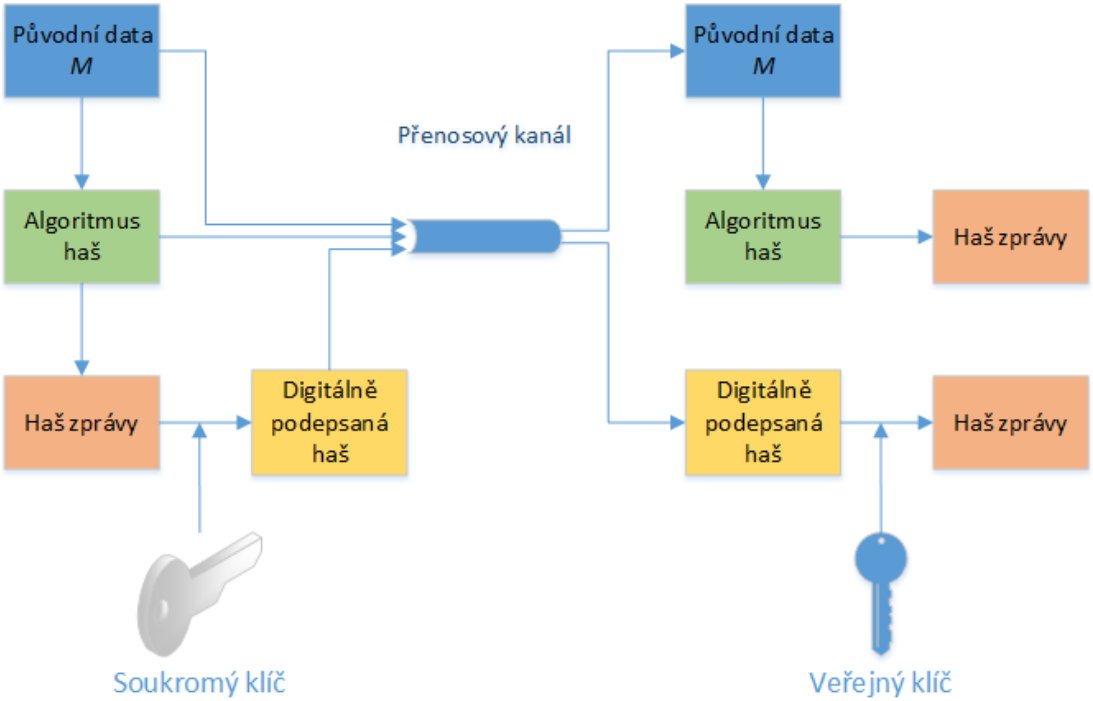
U asymetrické kryptografie jsou klíče odlišné a používá se pro ně pojem veřejný klíč a soukromý klíč. Soukromý klíč je tajný, zatímco veřejný klíč je zveřejněn komukoli. Veřejný klíč se používá k šifrování zprávy, což znamená, že každý uživatel může zprávu zašifrovat, ovšem pouze uživatel, který zná soukromý klíč, může takto zašifrovanou zprávu přečíst. Pomocí této metody lze zajistit utajení dat viz obr. 2.



Obr. 2 Princip šifrování pomocí asymetrické kryptografie

V případě, kdy uživatel na zprávu aplikuje soukromý klíč (podepisování zprávy) a příjemce veřejný klíč (ověření podpisu), hovoří se o takzvaném podpisovém schématu, viz obr. 3. Uživatel vlastnící soukromý klíč podepíše data, přičemž kdokoli je schopen ověřit validitu podpisu příslušným veřejným klíčem. Asymetrická kryptografie má klíče mnohem delší než symetrická kryptografie (1024, 2048 bitů). Proto šifrování a dešifrování zprávy trvá

delší dobu. Proto se počítá tzv. haš, což je jednocestná funkce (značí se H()), která ze zprávy libovolné délky vytvoří zprávu konstantní délky. Bezpečnost spočívá v tom, že je téměř nemožné najít dvě rozdílné zprávy se stejným hašem. Z vypočteného haše také není možné vypočítat zprávu původní. Vypočítá se tedy haš zprávy. Ten se podepíše a odešle společně se zprávou. Příjemce příjme data a vypočítá z nich haš. Takto získanou zprávu porovná s podepsaným přiloženým hašem. Pokud se tyto haše shodují, potom data nebyla nijak změněna a jsou považována za správná. Tomuto procesu se říká ověření podpisu. Jelikož se při podpisu využívá haš zprávy, je zajištěna také integrita dat.



Obr. 3 princip využití asymetrické kryptografie k vytvoření digitálního podpisu

Výhodou asymetrické kryptografie je, že veřejný klíč může být zveřejněn například na Internetu, kde si ho může stáhnout libovolný uživatel. Proto se v praxi využívají oba mechanismy současně. Symetrický pro šifrování a autentičnost dat a asymetrický pro autentizaci entit a sjednání symetrických klíčů.

Kryptografie není schopna zabezpečit absolutní bezpečnost. Není totiž schopna zajistit bezpečný hardware a software. Například bankovní transakce probíhají bezpečně, ale pokud existuje v počítači vir, který odešle útočníkovi přihlašovací údaje zadané z klávesnice, tak takový útočník může následně tyto transakce provádět. Pokud se navíc dokáže rozluštit používaný šifrovací algoritmus pomocí kryptoanalýzy, tak se tento algoritmus stává nebezpečným.

2 Autentizace

Autentizace slouží k určení, zda entita, se kterou je prováděna komunikace, je skutečně ta, za kterou se vydává. Autentizovat je možno osoby, zařízení či procesy. Autentizaci lze rozdělit na jednostrannou a oboustrannou. Při jednostranné autentizaci se autentizuje pouze jeden účastník. V případě plateb po Internetu, je třeba mít jistotu, že komunikace probíhá se serverem příslušné banky a nikoliv se serverem útočníka. V tomto případě se autentizoval server (typicky pomocí certifikátu). Opačný případ nastává u Wi-Fi, kdy se autentizuje klient (většinou pomocí hesla). Při oboustranné autentizaci se autentizují obě strany. Oboustranná autentizace je bezpečnější, nevýhodou však je, že server musí klienta znát. Existuje mnoho autentizačních metod, které budou popsány níže. V kapitole 2 bylo čerpáno z [5][6]

2.1 Autentizace znalostí

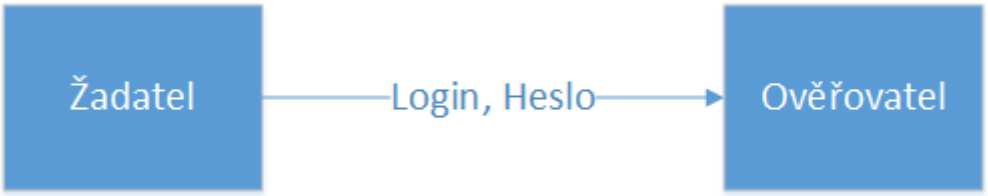
Při tomto druhu autentizace prokazuje entita svou identitu tím, že má nějakou znalost, kterou nemá nikdo jiný. Nevýhodou je, že aby byla taková autentizace bezpečná, musí být heslo dostatečně dlouhé a mělo by se skládat z pseudonáhodných znaků. Pokud je uživatelem člověk, musí být zvolené heslo dobře zapamatovatelné a relativně krátké. Pokud se ovšem zvolí nějaké dobře zapamatovatelné heslo, tak hrozí, že útočník využije tzv. slovníkový útok. Při tomto útoku útočník postupně zkouší hesla z připraveného slovníku nejpoužívanějších hesel. Tomuto útoku se lze bránit tak, že kontrolér umožní uživateli pouze omezený počet pokusů vložení hesla. Druhá možnost je ta, že kontrolér po každé neúspěšné autentizaci znemožní další pokus na určitý čas, který roste s počtem neúspěšných pokusů.

Pokud se tímto způsobem autentizuje přístroj s dostatečnou paměťovou kapacitou, tak může být heslo dlouhé a náhodné. Útočník by mohl použít tzv. útok hrubou silou neboli „brute force attack“. Ten spočívá ve zkoušení všech možných kombinací. Heslo musí být také nějakým způsobem uloženo. Útočník z něj pak může toto heslo získat. Proto se většinou ukládá pouze haš hesla. Kontrolér po přijetí hesla jednoduše spočítá haš a ověří jej s hodnotou uloženou v databázi, útočník přitom z tohoto haše není schopen zjistit příslušné heslo.

Dalším možným útokem je odposlechnutí hesla při přenosu. Útočník, který heslo získá, jej jednoduše použije pro příští autentizaci. Tomuto problému se dá zabránit pomocí autentizační metody výzva-odpověď.

2.1.1 Jednoduchá autentizace

Při tomto druhu autentizace se většinou používá login a heslo. Problémem tohoto druhu autentizace je, že není odolná vůči replay attacku. Někdy se navíc odesílají uživatelské informace (login a heslo) v nešifrované podobě, a tak je možné je odposlechnout. Tento typ autentizace se využívá například v protokolech FTP nebo TELNET (*TELecommunication NETwork*). Princip jednoduché autentizace je zobrazen na Obr. 4.

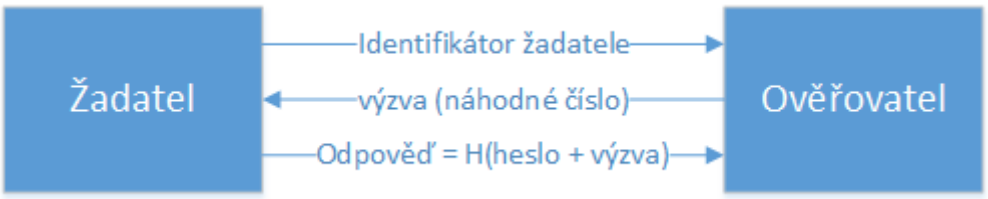


Obr. 4 Princip jednoduché autentizace pomocí jména a hesla

2.1.2 Metoda výzva odpověď

Při tomto druhu autentizace je komunikace mezi žadatelem a kontrolérem obousměrná. Žadatel zašle svoje údaje (například login). Kontrolér ověří, zda takového uživatele zná. Pokud ano, tak mu odešle náhodně vygenerovanou zprávu (výzvu). Uživatel zná své heslo a zná i tuto zprávu. Proto vypočte haš z řetězce heslo + výzva a takto vzniklou zprávu odešle kontroléru. Kontrolér vytvoří obdobným způsobem haš. Pokud jsou haše totožné, je uživatel autentizován, jelikož použil správné heslo. Demonstrace této metody je uvedena na obr. 5. Kromě využití hašovacích funkcí lze také využít symetrických šifrovacích algoritmů a asymetrických podpisových schémat pro konstrukci odpovědi.

U této metody je požadováno, aby výzva kontroléru byla pokaždé jiná. Pokud potom útočník zachytí zprávu *C*, nepovede se mu pomocí ní autentizovat, jelikož pro každou autentizaci je použita jiná výzva a k ní odpověď *C*. Z tohoto důvodu je součástí výzvy také časový záznam, který dělá výzvu jedinečnou a tedy odolnou vůči útokům opakováním tzv. replay attack.



Obr. 5 Princip autentizační metody výzva odpověď

2.2 Autentizace žadatelem

U této metody autentizace se žadatel prokazuje nějakými vlastnostmi, které jsou pro něj jedinečné. Výhodou této metody je, že žadatel si nemusí nic pamatovat, ani nemusí mít autentizační předmět. Nevýhodou je, že tyto metody neurčí uživatele absolutně jistě. Pokud uživatel v předchozím zadával heslo, musel je zadat přesně. Například pokud heslo bylo „abcd1234“ a uživatel zadal „abcd1233“, nebyl autentizován. U metody autentizace uživatelem tohle není možné, jelikož výsledky autentizace nebudou nikdy absolutně totožné s výsledky referenčními. Například pokud se uživatel ověřuje otiskem prstu, nevloží ho vždy do stejné polohy, a netlačí na detektor vždy stejnou silou. Vždy se tedy zkoumá jen míra pravděpodobnosti, zda je uživatel skutečně tím, za koho se vydává. Může tedy dojít k chybám. Existují dva druhy chyb. Buď bude povolen přístup neoprávněné osobě nebo bude odepřen přístup oprávněné osobě. Pravděpodobnost chyby prvního druhu je označena FAR (*False Acceptance Rate*), pravděpodobnost chyby druhého druhu pak FRR (*False Rejection Rate*). Snahou je dosáhnout co nejnižších hodnot FAR i FRR. To je problémem, jelikož snížení FAR souvisí se snížením rozhodovací úrovně a to naopak zvýší FRR.

Některé metody mohou být navíc uživateli nepříjemné. Autentizaci žadatelem můžeme dělit do dvou základních tříd:

2.2.1 Fyziologické metody

U těchto metod jsou zkoumány fyziologické proporce člověka, které žadatele jednoznačně identifikují. Nejčastěji se setkáváme s následujícími metodami:

- **Otisk prstů** – je v současné době hojně používaná metoda založená na tvaru papilárních linií.
- **Snímek obličeje** – je metoda srovnávající vytvořený snímek se snímkem uloženým v přístroji, případně v centrální ústředně.
- **Geometrie ruky** – přístroje tohoto typu měří délku a šířku prstů. Autentizace je rychlá, ale není moc spolehlivá.
- **Oční sítnice** – lidská sítnice obsahuje tenkou vrstvu buněk, které mění světlo na vzruchy posílané do mozku. Vrstva obsahuje cévy. Tvar těchto cév je pro každého člověka unikátní. Tato metoda může být pro uživatele nepříjemná a přístroje, které ji využívají, jsou drahé.
- **Duhovka** – při této metodě se hledají skvrny na lidské duhovce.
- **Cévní řečiště** – využívá unikátnosti lidského cévního řečiště. Měření probíhá na různých částech lidského těla. Nejpoužívanější je prst, dlaň nebo hřbet dlaně.

2.2.2 Behaviorální metody

U těchto metod se využívá lidského chování. V praxi jsou tyto metody méně časté než metody fyziologické.

- **Hlas –** využívá unikátních vlastností lidského hlasu jako je kadence nebo kmitočtové spektrum. Aby nebylo možné aplikovat tzv. replay attack, jsou některé přístroje vybaveny displejem a po uživateli vyžadují, aby říkal různá slova.

- **Psaní na klávesnici –** využívá charakteristiky typické pro uživatele, když píší na klávesnici. Například frekvenci úderů. Výhodou této metody je, že může uživatele autentizovat i při práci.

- **Psaný podpis –** u psaného textu lze měřit například sklon pera, tlak na podložku nebo tvar křivek.

Nevýhodou je, že některé tyto charakteristiky se mohou s časem měnit. Pokud si například uživatel ostříhá vlasy, nemusí být rozpoznán jeho obličej. V případě nemoci se může výrazně změnit lidský hlas atd. Útočník může také falšovat tyto údaje. Například si autorizovaného uživatele vyfotí nebo nahraje jeho hlas. I na to jsou některé kontroléry připraveny. Detektory obličeje mohou například zkoumat pohyb tváře, detektory hlasu mohou vyžadovat různá slova, která zobrazují na displeji a detektory otisků prstu zase proudění krve v prstu (testy živosti). Další nevýhodou je, že některé metody mohou být pro uživatele nepříjemné, například autentizace oční sítnicí.

2.3 Autentizace předmětem

Při této metodě prokazuje žadatel svou identitu tím, že vlastní jistý jedinečný předmět. Může se jednat o občanský průkaz, kartu, přívěsek, USB (*Universal Serial Bus*) token nebo mobilní telefon. Výhodou autentizace předmětem, je vysoká bezpečnost (předměty většinou mají velkou paměťovou kapacitu) a mnohdy také rychlost. Je například mnohem rychlejší přiložit kartu než zadávat osmimístné heslo. Nevýhodou je potom možnost odcizení předmětu a fakt, že žadatel musí mít předmět u sebe.

2.3.1 Karty

Karty jsou v současnosti velmi hojně využívány a to jak v přístupových systémech, tak i v sektoru bankovnictví. Informace o kartách byly čerpány převážně z literatury [6] a [7].

Karty lze obecně rozdělit do následujících skupin:

Karty s magnetickým proužkem

Informace zapsané do magnetického pásku, karta je lehce klonovatelná. Typickým příkladem je kopírování magnetického proužku u platebních karet, tzv. skimming.

Čipové karty

Nahradily karty magnetické, jelikož tyto karty byly náchylné na klonování. Navíc cena čipových karet velice klesla. Čipové karty rozdělujeme na:

- **Paměťové karty –** využívají se pouze pro autentizaci pomocí hesla (jednoduchá autentizace viz 2.1.1), jelikož umožňují pouze uchovávat data. Výhodou oproti magnetickým kartám je, že nejsou tak jednoduše klonovatelné.

- **Karty mikroprocesorové –** tyto karty poskytují vyšší bezpečnost. Přístup k datům může být zajištěn např. pinem a jejich cena rychle klesá. Tyto karty mohou realizovat různé výpočty (i kryptografické).

- **Karty kryptografické –** jsou podobné kartám mikroprocesorovým, mají ale navíc kryptografický procesor, který jim umožňuje efektivně vykonávat kryptografické operace.

- **Karty programovatelné -** existují různé typy karet. Nejznámější jsou Java Card, .NET karty (C#) a karty MULTOS (C). Tyto karty lze programovat, mají ovšem omezenou paměť a výpočetní výkon, což vede k využívání omezených API původních jazyků (Java ,C#).

2.3.2 Mobilní telefony

Mobilní telefony se používají jako autentizační předměty poměrně krátkou dobu. Výhodou použití mobilního telefonu jako autentizačního předmětu je, že v dnešní době vlastní mobilní telefon téměř každý a mnoho lidí jej nosí stále při sobě. Další výhodou je také fakt, že mobilní telefon má vyšší výpočetní výkon než čipové karty. To umožňuje využití vyšší úrovně zabezpečení (např. použití delších klíčů). Jistou nevýhodou je potom fakt, že data v mobilním telefonu nejsou tak bezpečně uložena jako například na čipových kartách. Přesto je využití mobilních telefonů jako autentizačních předmětů velice perspektivní, zejména ve spojení s technologií NFC, případně BLE. Mobilní telefon většinou nahrazuje právě čipové karty.

3 Technologie NFC

Technologie NFC (*Near Field Communication*) je moderní bezdrátová technologie, která vznikla v roce 2004. NFC umožňuje komunikaci na velmi krátkou vzdálenost (cca do 4 cm). Tato skutečnost má své výhody i nevýhody. Výhodou je například to, že taková komunikace je pro případné útočníky obtížně zachytitelná. Další výhodu lze nalézt například u využití NFC v oblasti karet. Nebylo by žádoucí, aby například v obchodě reagovala NFC čtečka na karty všech zákazníků v obchodě.

V současné době je tato technologie implementována do mnoha zařízení (i mobilních telefonů), a očekává se její velké rozšíření. Uplatnění této technologie lze najít na mnoha místech. Slouží například pro přenos kontaktů mezi mobilními telefony, v oblasti přístupových systémů, pro načtení zpráv z jídelního lístku, pro ochranu zboží, či pro bezkontaktní platby. V oblasti ochrany zboží nahrazuje NFC technologii RFID (*Radio Frequency IDentification*). V kapitole 3 bylo čerpáno převážně z [14][18].

Další přednost NFC je velmi krátká doba navázání spojení (méně než 100 ms). NFC pracuje na kmitočtu 13,56 MHz. Rychlost přenosu se pohybuje v rozmezí 106–424 kb/s (v závislosti na režimu komunikace), ale pracuje se již i na 1Mb/s verzi. Přenos je poloduplexní což znamená, že zařízení mohou buď přijímat, nebo vysílat data. Vždy tedy v jeden okamžik vysílá data pouze jedno zařízení. NFC zařízení může pracovat ve dvou módech aktivní a pasivní. V aktivním módu zařízení vysílá i přijímá data, vytváří tedy vlastní rádiové pole. Je tedy třeba, aby takové zařízení bylo napájeno. V pasivním režimu potom zařízení nevytváří vlastní radiové pole. Výhodou je, že takové zařízení nepotřebuje napájení, jelikož je napájeno aktivním zařízením, se kterým komunikuje.

Princip technologie NFC je prakticky stejný s principem technologie RFID. NFC je s RFID kompatibilní. RFID se využívá k identifikaci osob a k zabezpečení výrobků proti odcizení. Technologie pracuje s tzv. tagy. Tyto tagy jsou tvořeny cívkou, kondenzátorem a obvodem pro uložení unikátního čísla EPC (*Electronic Product Code*). Čtečka tagů vysílá impulzy, pomocí nichž nabije kondenzátor tagu. Jeho energie slouží k odeslání EPC. Nevýhodou této technologie je fakt, že komunikace probíhá pouze jednosměrně. Tento nedostatek je již u NFC odstraněn.

3.1 Módy NFC komunikace

Přenos může probíhat ve třech módech. Jsou to módy pro čtení a zápis (Read/Write), mód rovný s rovným (Peer to Peer) a mód emulátoru karty (Card Emulation).

Mód pro čtení a zápis

V tomto módu komunikace je vždy jeden prvek aktivní a druhý pasivní viz obr. 6. Komunikace spočívá pouze v čtení nebo zápisu informací z nebo do pasivního čipu. Přenosová rychlost v tomto módu je 106 kb/s. Uplatnění tohoto módu je například v sektoru ochrany zboží. V tomto módu zařízení pouze čtou NDEF *(NFC Data Exchange Format*) zprávy z NFC tagu nebo do něj naopak zapisují data. Tento mód je nejjednodušší a byl přístupný již v zařízení s verzí Androidu 2.3 (API (*Application Programming Interface*) verze 9). Zařízení pracující v tomto jsou kompatibilní s čipovými kartami dle standardu ISO (*International Organization for Standardization*)/IEC (*International Electrotechnical Commission*) 14443.

Obr. 6 NFC Mód pro čtení a zápis.

Mód rovný s rovným

V tomto módu spolu komunikují dvě aktivní NFC zařízení viz obr. 7. Přenosová rychlost dosahuje 424 kb/s. Tento mód se uplatňuje například při výměně kontaktů mezi mobilními telefony. Operační systém Android k této operaci využívá tzv. Android beam. Tato funkce umožňuje zaslat NDEF zprávy z jednoho zařízení do druhého. Aktivace se provádí fyzickým přiblížením dvou aktivních NFC zařízení.

Mód je podporován od verze Androidu 2.3 (API verze 9).

Obr. 7 NFC mód pro rovný s rovným.

Mód emulátoru karet

V tomto módu komunikují dvě aktivní zařízení, jedno z nich se ovšem chová jako pasivní, kdy simuluje čipovou kartu, viz obr. 7. Takové zařízení je velmi často mobilní telefon, čehož lze využít např. při bezkontaktních elektronických platbách, fyzických přístupových systémech apod. Telefon má totiž podstatně vyšší výpočetní výkon než pasivní tag, proto lze v tomto módu komunikaci šifrovat, což je velice důležité. Takto lze komunikovat buď s jiným Android zařízením, který se chová jako čtečka, případně s hardwarovou čtečkou. Tento mód je přístupný až od verze Androidu 4.4 (API veze 19).

Obr. 8 NFC mód emulátor karet

3.2 NFC v operačním systému Android

Zařízení, která využívají operačního systému Android s potřebnou API a jsou vybaveny hardwarovou podporou NFC mohou tuto technologii využívat prostřednictvím aplikací spuštěných na telefonu. Tato funkcionalita je podporována od verze 2.3 API verze 9. Zařízení komunikují pomocí tzv. NFC tagů. Tyto tagy v sobě obsahují data ve formě zpráv. Tyto zprávy mohou mít různý formát, ovšem v Androidu se velmi často pracuje se zprávami v NDEF formátu.

V rámci této kapitoly je popsán samotný vývoj aplikací pro OS Android. Tato kapitola přepokládá základní znalosti programování pro OS Android, které lze případně získat z [1][13], pro získání znalostí o programování v jazyce Java, na kterém je vývoj založen je doporučena literatura [11]. V následujících kapitolách bylo převážně čerpáno z [2].

Technologie NFC byla do Androidu implementováno od verze 2.3 (API verze 9). Zde byl ovšem k dispozici pouze omezený tag dispatch systém (viz níže). Verze 2.3.3 (API verze 10) již obsahuje kompletní možnost I/O (I*nput/Output*) operací a verze 4.0 (API verze 14) nabízí nové metody pro vysílání zpráv do jiných zařízení pomocí Andoid Beam.

Aby bylo zajištěno, že aplikace nebude spuštěna na zařízeních, která nemají vhodnou verzi systému, musí se uvést do manifestu zapsat kód:

```
<uses-sdk android: minSDKVersion = "14">
```

Příklad neumožní instalaci na zařízení s API verzí nižší než 14 (4.0). Je ovšem důležité uvědomit si, že ne každé zařízení (i s API verzí větší než 14) podporuje NFC. Aby byla aplikace nainstalována pouze na zařízení, které NFC podporují, je třeba do manifestu napsat následující kód:

```
uses-feature android: name= "android.hardware.nfc" android: required = "true">
```

`android: required` se nastavuje na true, pokud je práce s NFC pro aplikaci nutná. Pokud bude tato hodnota nastavena, nebudou zařízení bez hardwarové podpory NFC schopna aplikaci nainstalovat.

Pro práci s NFC je třeba sdělit systému, že aplikace bude s touto technologií pracovat. Uživatel potom při instalaci potvrdí, že je seznámen s tím, že příslušná aplikace technologii NFC využívá a bude k ní mít přístup. Je tedy třeba uvést:

```
<uses-permission android:name="android.permission.NFC" />
```

3.3 Formáty zpráv

Android obsahuje velké množství knihoven pro práci s NDEF zprávami. Lze používat i zprávy jiné, ale psaní aplikace bude pak složitější. V takovém případě je třeba, aby si programátor definoval zprávy sám v rámci vlastního protokolu.

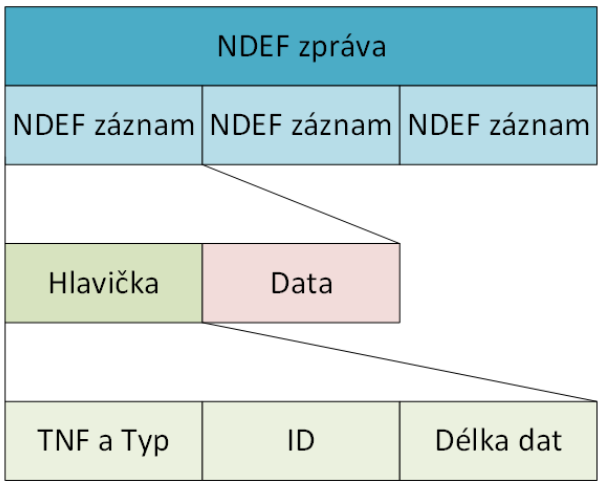
3.3.1 NDEF zprávy

V této sekci je popsáno, jak vytvářet NFC komunikaci pomocí NDEF zpráv. Jedná se o standartní formát NFC zprávy. Využívá se v módu read/write a peer to peer.

Struktura NDEF zprávy je zobrazena na Obr. 9. Komunikace se dělí na dvě fáze: Čtení dat z NFC tagu a vysílání NDEF zpráv. Význam jednotlivých polí ve zprávě je následující:

- TNF (*Type Name Field*) – jedná se o tříbitový identifikátor, který určuje, co obsahuje pole typ (bod b).
- Typ proměnné délky – určuje typ záznamu.
- ID (*IDentifier*) – unikátní číslo záznamu, často se nepoužívá.
- Délka dat – délka dat, která jsou v aktuálním záznamu přenášena.
- Data proměnné délky – data, která pro nás mají význam. Jelikož zpráva může obsahovat více záznamů, nemusí všechna data být v prvním z nich.

První tři z uvedených jsou obsaženy v hlavičce záznamu, viz obr. 9.



Obr. 9 Struktura NDEF zprávy

Čtení dat

O čtení NDEF zpráv z NFC tagu se stará tzv. tag dispatch systém. Ten zachytí příchozí tag, analyzuje jej a osloví příslušnou aktivitu (která má zaregistrovaný příslušný intent filter).

Vysílání NDEF zpráv

Vysílání zajišťuje tzv. AndroidBeam. Výhodou oproti jiným bezdrátovým systémům jako Wi-Fi nebo bluetooth je to, že NCF komunikace nevyžaduje žádné párování.

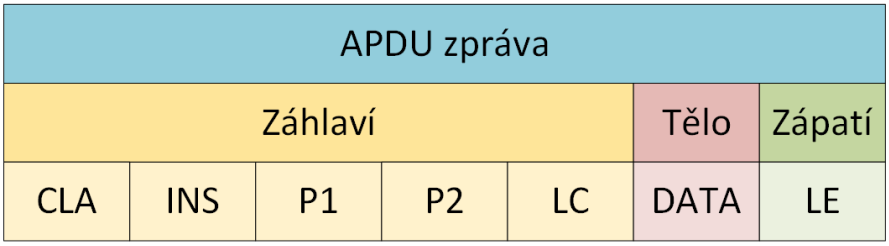
3.3.2 APDU zprávy

Při komunikaci pomocí emulátoru karet se používají standardní zprávy tzv. APDU (*Application Protocol Data Unit*) zprávy. Struktura APDU zprávy se všemi využitými poli je zobrazena na obr. 10 Zpráva vyslaná do karty obsahuje následující pole:

- CLA (*Class of Instruction*) – jedná se o instrukční třídu. Pokud je nastavena na hodnotu 00, definuje, že APDU zpráva slouží k výběru aplikace. Hodnota 80 potom znamená komunikaci s danou aplikací.
- INS (*Instruction*) – specifikuje konkrétní operaci. Může po kartě vyžadovat například výsled nějakého výpočtu, případně vrácení určitých dat.
- P1 – parametr specifikující příkaz
- P2 – parametr specifikující příkaz

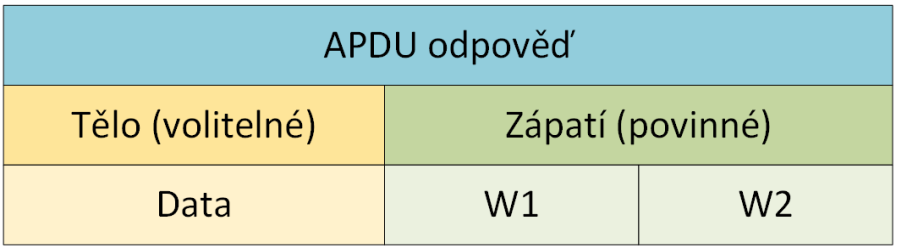
Další položky zprávy jsou volitelné. Patří mezi ně:

- LE (*Length of the Expected response*) – určuje maximální délku zprávy, kterou očekává terminál od karty.
- LC (*Lenght of the Command data*) – délka dat, jenž jsou odeslána ve zprávě
- Data – samotná přenášená data.



Obr. 10 Struktura APDU zprávy

Struktura odpovědi karty (APDU Response) je patrná z obr. 11. Odpověď je složena z nepovinného těla, obsahující přenášená data z karty a povinného zápatí, obsahující návratové kódy W1 a W2, indikující výsledek provedení operací na kartě dle specifikace ISO 7816-4. V případě, že dojde k úspěšnému zpracování dat na kartě, jsou návratové kódy ve tvaru W1=0x90 a W2=0x00.



Obr. 11 Struktura APDU odpovědi

3.3.3 Intenty

Intenty jsou zprávy, které se posílají uvnitř zařízení s OS Android. Vyjadřuje operaci, která má být vykonána. Používá se například pro spouštění nových aktivit. U každé aktivity je uveden intent filter, který určuje, na jaký typ intentu příslušná aplikace reaguje.

Pokud nastane nějaká systémová událost, upozorní na ni systém právě vysíláním intentu. Pokud je na tento intent registrováno více aktivit, bude uživateli umožněn výběr z jejich výčtu. Intent je například vyslán při spuštění hudby uživatelem.

3.3.4 Ostatní typy zpráv a technologií

Přestože NDEF zprávy jsou v Androidu nejlépe podporovány a nejvíce rozšířeny, lze použít i formáty jiné. Podporované technologie lze získat pomocí metody `getTechList()`. Patří mezi ně NfcA, NfcB, NfcF, NfcV a IsoDep viz [2]. Práce s těmito technologiemi obnáší registrovat aplikaci tak, aby očekávala akci ACTION_TECH_DISCOVERED. Dále je možné použít výčet podporovaných technologií pomocí XML (*eXtensible Markup Language*) souboru. Z příchozího inentu lze vytvořit tag, a to pomocí následujícího volání:

```
Tag tagFromIntent = intent.getParcelableExtra(NfcAdapter.EXTRA_TAG);
```

Také lze získat instanci použité technologie a to s využitím výše vytvořeného tagu. Zde je uveden příklad pro technologii IsoDep:

```
IsoDep myTag = IsoDep.get(tagFromIntent);
```

3.4 Vývoj aplikací peer to peer

Zařízení očekávají NFC komunikaci pokud je obrazovka odemčená. V opačném případě nebudou schopny komunikovat. Při vývoji aplikace je důležité, aby komunikace probíhala nejlépe bez zásahu uživatele, jelikož vlivem své činnosti by uživatel mohl zařízení vzdálit od jeho protějšku natolik, že by nebylo možné dále komunikovat. K tomu může dojít například v případě, kdy uživatel přiloží své zařízení k čtečce. Následně je dotázán na volbu aplikace, která má být spuštěna (reaguje na příslušný intent). Po této výzvě uživatel nevědomky zařízení od čtečky vzdálí a komunikace se tím ukončí. Proto je třeba se snažit, aby byla aplikace vybrána automaticky. K tomuto účelu složí výše zmíněný tag dispatch system.
Ten postupuje následovně:

- získá MIME (*Multipurpose Internet Mail Extensions*) typ nebo URI (*Uniform Resource Identifier*) ze zprávy, tedy identifikátory aplikace
- zapouzdří tuto informaci společně s daty do intentu
- tento intent předá příslušné aktivitě.

3.4.1 Tag dispatch systém

Tag dispatch systém slouží ke správnému určení aplikace, které jsou data určena. Snaží se tak zabránit zbytečné interakci s uživatelem. Přiřazení jednotlivých aplikací je možné pouze

pokud jsou data ve formátu NDEF zpráv. Ta obsahuje jeden nebo více NDEF záznamů. Tyto záznamy mají speciální formát a používají se k přiřazení MIME typu nebo URI.

Ať jsou data v jakémkoli formátu, vždy jsou vytvořeny intenty. Tyto intenty mají různou prioritu. Největší prioritu má intent s akcí `ACTION_NDEF_DISCOVERED`. Pokud je zpráva ve formátu NDEF, a nějaká aplikace je registrována na tuto akci, bude spuštěna právě tato aplikace. Pokud by takových aplikací bylo více, zobrazí se uživateli výběr z těchto aplikací. Pokud zpráva není ve formátu NDEF nebo žádná aplikace nemá registrovánu akci `ACTION_NDEF_DISCOVERED`, vyvolá tag dispatch system intent s akcí `ACTION_TECH_DISCOVERED`. Pokud je registrováno více aplikací, platí stejná pravidla jako u `ACTION_NDEF_DISCOVERED`. Pokud ani na tuto akci žádná aplikace nereaguje, je zde poslední možnost: `ACTION_TAG_DISCOVERED`. Pokud ani nyní nenajde žádnou shodu, skončí. Z tohoto důvodu je dobré registrovat své aplikace na `ACTION_NDEF_DISCOVERED`, pokud používáme NDEF formát zpráv. V jiném případě je vhodné používat akci `ACTION_TECH_DISCOVERED`, protože aplikace s `ACTION_TAG_DISCOVERED` mají nejmenší prioritu a nemusí na ně vůbec dojít.

3.4.2 Android beam

Android Beam umožňuje komunikaci mezi dvěma zařízeními s OS Android, ale pouze pokud komunikace probíhá pomocí NDEF zpráv. Pokud chce aplikace komunikovat, musí být v aktivním stavu a druhé zařízení nesmí být zamčeno. Existují dva možné přístupy:

Metoda `setNdefPushMessage()` – přijímá jako parametr objekt typu `NdefMessage`. Tato zpráva bude pak automaticky vyslána, když se zařízení přiblíží k sobě. Tato metoda se tedy používá tehdy, pokud chceme vyslat vždy stejná data.

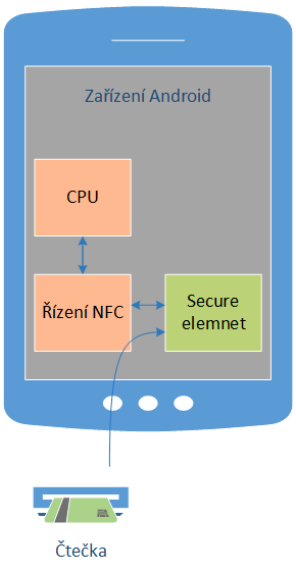
Pomocí `setNdefPushMessageCallback()` – pokud se zařízení přiblíží, je vyvolána metoda `createNdefMessage()` z callbacku, který byl registrován. To umožňuje aplikaci rozhodovat se, zda má či nemá nějaká data odeslat.

3.4.3 Aplikační záznamy

Od verze 4.0 (API verze 14) podporuje Android aplikační záznamy, aby se zajistila komunikace se správnou aplikací a nemuselo se spoléhat pouze na intenty. Pokud systém nenajde aplikaci s příslušným záznamem, spustí se vyhledávání aplikace pomocí Google Play. Jelikož tyto záznamy jsou dostupné ad API verze 14, je třeba pro podporu starších zařízení implementovat i intent filter.

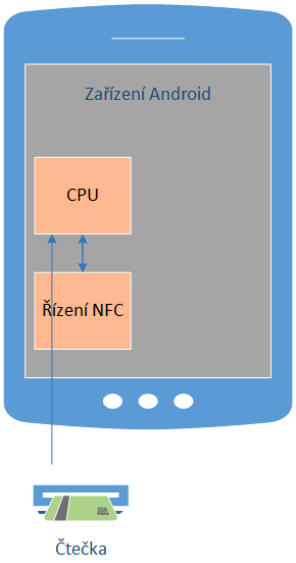
3.5 Emulátor karet

Pokud má zařízení komunikovat se čtečkou karet, musí být vybaveno příslušným NFC hardwarem. Pro zajištění bezpečnosti zde slouží secure element. V případě, že komunikující zařízení má API úroveň nižší než 4.4, zpřístupní aplikace pouze secure element a komunikace dále probíhá pouze mezi čtečkou a secure elementem viz obr. 12. Po skončení přenosu se může aplikace dotázat secure elementu na výsledek přenosu a sdělit ho uživateli. Aplikace jako taková se na NFC komunikaci nepodílí.



Obr. 12 Znázornění NFC komunikace mezi čtečkou a secure elementem

V API 4.4 přibyla nová možnost komunikace a to bez secure elementu. Čtečka v tomto případě komunikuje přímo s aplikací. Toto schéma je k vidění na obr. 13



Obr. 13 Komunikace NFC čtečku a zařízením bez použití secure elementu

Android 4.4 podporuje simulaci karet s technologií IsoDep, založené na APDU zprávách, definovaných v ISO/IEC 7816-4 standardu. Komunikace se zařízením probíhá tedy stejně jako by se jednalo o klasickou čipovou kartu. Jedno zařízení vystupuje jako klient (terminál/čtečka) a druhé jako server (čipová karta).

3.5.1 Vývoj aplikací HCE

K účelu emulování karet na mobilním telefonu s operačním systémem Android se využívá HCE (*Host Card Emulation*). Aplikace je spuštěna na zařízení jako služba (třída `Service`). Této třídy se zde vyžívá proto, že nepotřebuje interagovat s uživatelem pomocí uživatelského rozhraní, což je funkcionalita, jaká je od karty očekávána.

Výběr příslušné aplikace

Po přiložení zařízení ke čtečce, je třeba identifikovat příslušný emulátor karet, jelikož v zařízení může existovat více virtuálních karet. K tomuto účelu slouží tzv. AID (*Application IDentifier*). AID je 16 bitový identifikátor. U již existujících systémů jsou AID definovaná a veřejně dostupná. Při programování aplikací pro takové systémy je tedy třeba příslušné AID zjistit. Takovým známým systémem může být například Visa nebo MasterCard. Pokud bychom chtěli navrhnout vlastní systém čteček, bylo by třeba své AID registrovat.

Někdy je třeba registrovat jedné kartě více AID. K tomu slouží AID skupiny. Android garantuje, že karta bude reagovat na všechny nebo žádné AID ze skupiny. Těmto skupinám je možné přidělit kategorii a to buď `CATEGORY_PAYMENT` nebo `CATEGORY_OTHER`. Jak již z názvu vyplývá, první kategorie slouží pro platební karty a druhá pro ostatní.

Oprávnění a ověření dostupnosti emulátoru

Když je zapotřebí pracovat s emulátorem karet, je potřeba do manifestu projektu uvést pravidlo povolující technologii NFC:

```
<uses-permission android:name="android.permission.NFC"/>
```

a také pravidlo povolující práci služby emulátoru karet:

```
<uses-feature android:name= FEATURE_NFC_HOST_CARD_EMULATION />
```

Tento kód zajistí, že aplikace nebude spuštěna na zařízení, které nepodporuje emulaci karet.

Implementace služby

Služba pro emulátor karet dědí ze třídy `HostApduService`, která obsahuje dvě abstraktní metody. Jsou to `processCommandApdu()`, která je volána, když zařízení příjme data od čtečky. Odpověď lze vrátit přímo z této metody, případně lze vrátit null a odpověď posílat

později pomocí `sendResponseApdu()`. Druhou abstraktní metodou je `onDeactivated()`, která se volá po ukončení komunikace. K tomu dojde, když uživatel vzdálí kartu od čtečky, nebo když čtečka pošle novou zprávu s informací o AID (výběr jiné HCE (*Host-based Card Emulation*) aplikace). V této metodě lze zjistit, z jakého z těchto důvodů k ukončení došlo.

Tato služba, stejně jako každá jiná, musí být uvedena v manifestu. Musí ovšem obsahovat některé nestandardní prvky:

- Intent filter s akcí

```
<action
android:name=”android.nfc.cardemulator.action.HOST_APDU_SERVICE”>
```

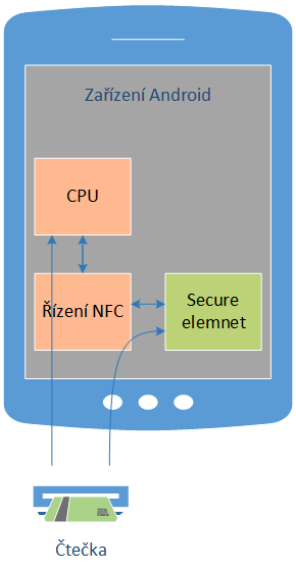
- Element `<meta-data>` ukazující na soubor s informacemi o službě. Může se jednat například o AID filtr, AID skupina, popis a další.
- Nastavit hodnotu `android:exported="true"`
- Vyžádat oprávnění

```
android:permision ="android.permission.BIND_NFC_SERVICE"
```

Toto pravidlo povolí spojení služby s externími aplikacemi.

Spolupráce se secure elementem

Spolupráce probíhá pomocí směrování AID zpráv. Čtečka komunikuje s NFC kontrolérem, který uchovává směrovací tabulku, která obsahuje AID a příslušného příjemce. Tímto příjemcem může být CPU (*Central Processing Unit*), kde je spuštěna aplikace, nebo secure element viz obr. 14. Pokud přijde ze čtečky první zpráva SELECT_AID, kontrolér vyhledá tento záznam ve své tabulce. Tímto směrem posílá zprávy až do doby, kdy se spojení rozpadne nebo je poslána nová SELECT_AID. Pokud kontrolér nenajde příslušnou AID, může použít defaultní cestu. Lze implementovat i komunikaci, bez zásahu aplikace, jak tomu bývalo v dřívějších verzích Androidu, je třeba pouze udělat pár změn v manifestu.



Obr. 14 Sdílení NFC komunakace mezi aplikace a secure element. Zprávy s AID aplikací jsou směrovány aplikacím (levá modrá šipka), ostatní potom secure elementu.

3.5.2 Bezpečnost

Jelikož je ve službě vyžadováno povolení `BIND_NFC_SERVICE`, je zaručeno, že zprávy projdou kontrolérem, a to tam i zpět. Není zaručen pouze původ zpráv. Pro zlepšení bezpečnosti je možné využít například aplikaci sandbox, která izoluje data od dat jiných aplikací. Bezpečnost je do jisté míry zajištěna samotným NFC, jelikož díky krátkému dosahu je komunikace těžko odposlechnutelná.

4 Technologie BLE

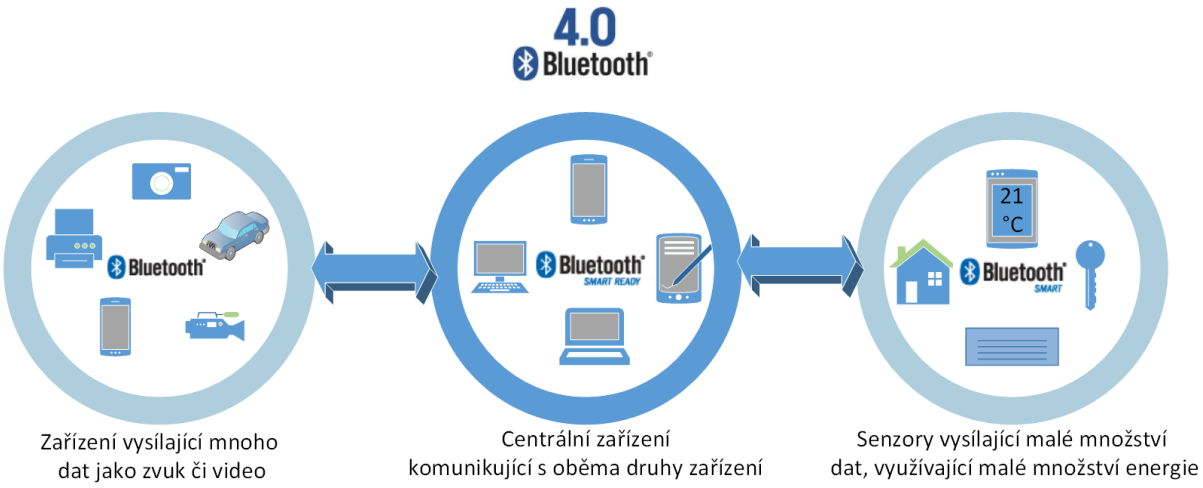
BLE (*Bluetooth Low Energy*) je moderní komunikační technologie (někdy také nazývána jako bluetooth smart). Tato technologie byla vydána ve verzi bluetooth 4.0. Výhodou oproti předchozím verzím je, jak již název napovídá nízká spotřeba energie. Obyčejná plochá baterie může napájet zařízení komunikující pomocí BLE až po dobu tří let. Další výhodou je nízká cena. Technologie bluetooth je navíc již nyní velice oblíbena u výrobců. Z tohoto důvodu je to technologie ověřená. Dosah BLE je až 100 metrů, což je pro většinu aplikací postačující. Na BLE je také založena technologie iBeacons od společnosti Apple. Tato technologie umožňuje mobilním telefonům (tzv. iBeacons) naslouchat zařízením (tzv. beacons) a komunikovat s nimi. Chytré telefony tak mohou zjišťovat pozici vhledem k ostatním zařízením pracujícím na této technologii. V této kapitole bylo čerpáno z [3], [16] a [2].

BLE pracuje na frekvenci 2,4 GHz. Podporuje odesílání velmi krátkých zpráv. Nejkratší možná zpráva má délku 8 bajtů, nejdelší potom 27 bajtů. Přenosová rychlost je 1Mb/s. Aby bylo sníženo rušení jiných zařízení, používá BLE techniky adaptivního frekvenčního skákání, kdy na základě kvality jednotlivých kanálů, může zařízení měnit svůj vysílací kanál. Šetření energie spočívá v přenesení většiny práce do řídícího zařízení (master). Proto zařízení periferní (slave) mohou být po delší dobu v režimu šetření baterie. Řídící zařízení potom v předem definovaných časových intervalech očekává data od svých periferních zařízení, která tedy za velmi krátký čas vysílají a poté se opět přepnou do úsporného režimu. BLE využívá 24 bitový CRC (*Cyclic Redundancy Check*) pro odstranění chyb, které mohou při přenosu dat vzniknout. Každý paket obsahuje unikátní 32 bitový identifikátor, který jedinečně identifikuje příslušné periferní zařízení.

Technologie BLE má v praxi obrovské využití v mnoha oborech. Přesto, že je tato technologie relativně nová, je již integrována v mnoha zařízeních. Zejména ji využívají různé fitness doplňky jako měřiče tepu, krokoměry a podobné. V budoucnu se předpokládá integrace do mnoha věcí a tím naplnit koncepci internetu věcí (Internet of Things). Lidé budou například moci zamykat a odemykat auta pomocí jejich chytrých telefonů, ovládat svoji televizi, ledničku, hudbu do sluchátek nebo osvětlení domu. Rozdělení zařízení lze vidět na obr. 15.

Zařízení v BLE může vystupovat buď jako **centrální** (central) nebo jako **periferní** (peripheral). Periferní zařízení jsou taková, která pouze zasílají informace o tom, že existují. Nejsou tedy schopny hledat ostatní zařízení. Centrální zařízení jsou potom taková, která tato periferní zařízení hledají a navazují s nimi spojení. Z toho důvodu nemohou komunikovat ani dvě periferní, ani dvě centrální zařízení společně. Centrální i periferní zařízení mohou pracovat ve dvou stavech, a to klient nebo server. Typičtější je, když periferní zařízení pracuje jako server. Tato situace například nastane, pokud jsou vyžadována nějaká data od chytrých

hodinek řízených BLE. Opačná situace (kdy centrální zařízení vystupuje jako sever a periferní jako klient) nastává v případě, že chceme do našich hodinek nahrát nová data.



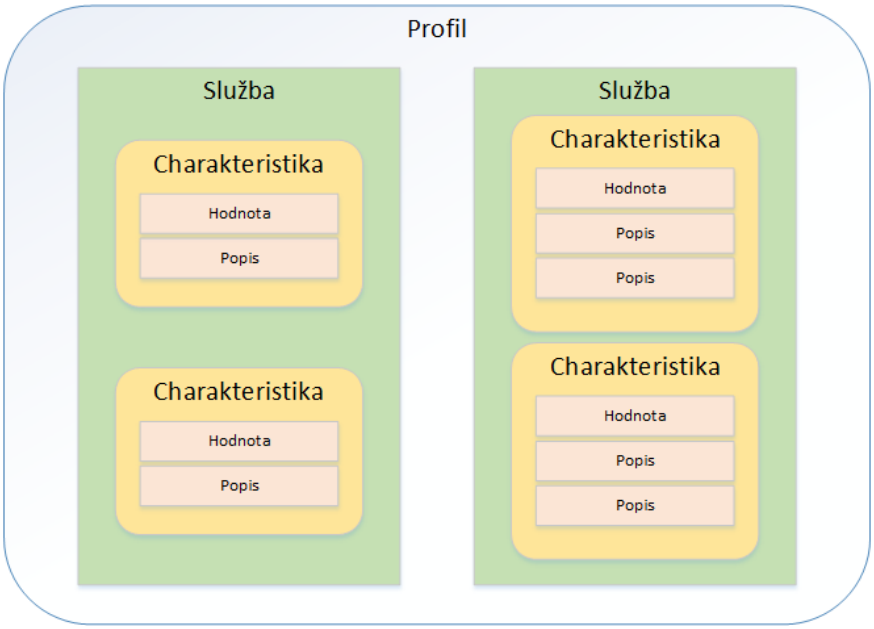
Obr. 15 Rozdělení zařízení v BLE.

4.1 BLE v OS Android

Android podporuje BLE od verze 4.3 (API verze 18). V této verzi ovšem umožňuje zařízením vystupovat pouze jako centrální zařízení (central). Jako taková dovedou hledat a připojovat tzv. periferní zařízení (peripherals). Následně je schopno z těchto zařízení získávat data, případně do nich data ukládat. Od verze Android 5.0 (API verze 21) je možné, aby zařízení vystupovalo jako periferní zařízení.

Android využívá ke komunikaci pomocí BLE speciální profil tzv. GATT (*General Agreement on Tariffs and Trade*). Každé zařízení může využívat více profilů založených na profilu GATT pro komunikaci s různými zařízeními. Příklad profilu založeném na GATT lze vidět na obr. 16. Při BLE komunikaci jsou vysílány krátké zprávy, tzv. atributy. Tyto atributy mají unikátní 128 bitový identifikátor UUID (*Universally Unique IDentifier*). Atribut může být formován jako služba nebo jako charakteristika.

- **Charakteristika** – obsahuje hodnotu a žádný nebo několik popisů (descriptors). Data jsou ta část, která je pro nás důležitá. Může se jednat například o čas z hodinek, se kterými pomocí BLE komunikujeme. Popis potom poskytuje informaci o tom, v jakém formátu je hodnota napsána. V případě hodinek například jistý formátovací řetězec.
- **Služba** – je sdružení jedné nebo více charakteristik dohromady. Slouží ke sloučení charakteristik patřících do jisté logické skupiny.



Obr. 16 Struktura BLE profilu

4.1.1 Prerekvizity programování BLE

BLE je v Androidu podporováno až od verze 4.3 (API verze 18). Seznam zařízení s hardwarovou podporou BLE lze najít na http://www.bluetooth.com/Pages/Product-Directory.aspx . Při vývoji aplikací využívající technologii BLE, je nejdříve nutné tuto technologii povolit. Proto je třeba do manifestu projektu napsat následující pravidlo:

```
<uses-
featureandroid:name="android.hardware.bluetooth_le"android:required=
"true"/>
```

Tento zápis neumožní nainstalovat aplikaci na zařízení, které nemají hardwarovou podporu BLE. Zda zařízení podporuje BLE lze zjistit i za běhu programu pomocí následujícího kódu:

```
(getPackageManager().hasSystemFeature(PackageManager.FEATURE_BLUETOO
TH_LE)
```

Aby se zamezilo instalaci aplikace na zařízení s nízkou verzí Androidu, použije se následující kód:

```
    <uses-sdk android: minSDKVersion = "18">
```

Pokud je třeba, aby zařízení vystupovalo v roli periferie, bylo by zapotřebí místo verze API 18 uvést verzi API 21 (Android 5.0).

Poslední věc, kterou je potřeba provést je vyžádat si příslušná práva.

```
<uses-permission android:name="android.permission.BLUETOOTH"/>
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
```

Kde první pravidlo povoluje aplikaci komunikaci pomocí BLE (příjem dat, přijímání dat, párování). Druhé pravidlo je pak nepovinné a použije se pouze v případě, že se požaduje pomocí psané aplikace hledat jiné zařízení s BLE nebo měnit nastavení BLE.

4.1.2 Vývoj aplikací s podporou BLE v OS Android

Nejdříve je potřeba získat objekt `BluetoothAdapter`. Tento objekt representuje BLE adaptér zařízení. Pomocí tohoto objektu tedy probíhá komunikace s fyzickým adaptérem.

Objekt `BluetoothManager` se získá pomocí následujícího kódu:

```
BluetoothManager manager = (BluetoothManager)
getSystemService(BLUETOOTH_SERVICE);
BluetoothAdapter adapter = manager.getAdapter();
```

Ověření, zda je BLE povoleno se provede pomocí metody `isEnabled()` objektu `BluetoothAdapter`.

Vyhledávání BLE zařízení

Pokud zařízení vystupuje v centrální roli, může hledat jiná BLE zařízení. K tomu slouží metoda `startLeScan()`.Této metodě se předá `BluetoothAdapter.LeScanCallback`, který se vyvolá po získání výsledků. Pokud bychom chtěli hledat pouze určitá periferní zařízení, můžeme použít metodu `startLeScan(UUID[]`. Jelikož hledání zařízení je energeticky náročné, je potřeba skončit s vyhledáváním hned, když nalezneme příslušné zařízení.

Připojení k vyhledanému zařízení

Po vyhledání příslušného zařízení je třeba se k němu připojit. Konkrétně ke GATT serveru zařízení. K tomu slouží metoda `connectGatt()` třídy `BluetoothDevice`. Tato metoda přijímá jako parametr objekt typu `BluetoothGattCallback` .V tomto callbacku můžeme ověřit spojení a zjistit další informace o spojení. Metoda `connectGatt()` vrací objekt typu `BluetoothGat`, který slouží k provádění různých klientských operací, například umožňuje zjistit, jaké služby zařízení nabízí.

Příjem a odesílání BLE atributů

Po připojení ke GATT serveru je aplikace schopna přijímat a odesílat atributy. Pro přístup ke službám daného zařízení slouží metoda `getServices()` objektu `BluetoothGatt`.Ta vrací list služeb podporovaných daným zařízením. Pro výběr pouze příslušné služby slouží metoda `getService(UUID uuid)` téhož objektu, kde UUID definuje příslušnou službu. K jednotlivým charakteristikám lze potom přistoupit pomocí metody

`getCharacteristics()` objektu `BluetoothGattService`. Pro získání pouze určité charakteristiky slouží metoda `getCharacteristic(UUID uuid)`. Ta pracuje stejně jako v případě služeb. Jak již bylo uvedeno výše, každá charakteristika obsahuje data a volitelně jeden nebo více popisů. Data se získávají z charakteristiky pomocí metody `getValue()` a popisy pomocí metody `getDescriptors()`. Aplikace může reagovat na změnu nějaké charakteristiky, voláním metody `setCharacteristicNotification()` třídy `BluetoothGat`.Pokud se potom příslušná charakteristika změní, bude vyvolána metoda o`nCharacteristicChanged()`. Ve které je možno na tyto změny reagovat.

Ukončení práce s BLE

Po ukončení práce s BLE je potřeba uvolnit zdroje, aby je bylo možné použít v jiných aplikacích. Toho lze dosáhnout pomocí metody `close()` objektu `BluetoothGat` .

5 Bezpečné uložení klíčů

Bezpečnostní klíče jsou součástí mnoha kryptografických protokolů. Z důvodu bezpečnosti je ovšem nutné některé klíče uchovávat tak, aby s nimi nemohl neoprávněný uživatel manipulovat. Jedná se o tajné klíče symetrické kryptografie, soukromé klíče asymetrické kryptografie a certifikáty. Programovací jazyk Java i operační systém Android umožnují bezpečné uložení těchto informací.

5.1 Keystore a Keychain v OS Android

Android poskytuje dvě možnosti uložení tajných dat. Lze to provádět pomocí Android keystore a Android keychain. Při použití kteréhokoliv z nich je zaručeno, že data nebude možné (nebo velice komplikované) odebrat ze zařízení. Rozdíl mezi těmito dvěma přístupy je ten, že android keychain uchovává data na úrovni zařízení, čili tato data mohou být přístupná více aplikacím. Android keystore naopak uchovává data přiřazené jedné aplikaci a žádná jiná aplikace nemá k těmto datům přístup. Objekt třídy keystore obdržíme následujícím voláním:

```
KeyStore ks = KeyStore.getInstance(String type);
```

Nejvýznamnější hodnoty parametru type:

- JKS (*Java KeyStore*) – Keystore získaný touto metodou umožnuje uložit pouze privátní klíče a certifikáty, nikoli však tajné klíče symetrické kryptografie. Data jsou dostupná pouze v rámci programovacího jazyka Java. Tento formát je výchozí hodnotou.
- JCEKS (*Java Cryptography Extension Keystore*) – je rozšířením JKS. Může například uchovávat i tajné klíče symetrické kryptografie. Uchovávaná data jsou také bezpečněji uloženo pomocí algoritmu triple DES (*Data Encryption Standard*).
- PKCS12 (*Public-Key Cryptography Standards 12 Keystore*) – kdy získaný objekt může být použit napříč více programovacími jazyky jako například C, C++ nebo C#.
- BKS (*Bouncy Castle Keystore*) – je speciální typ keystore užívaný v Bouncy Castle verzích JDK.

Data jsou representována objekty `KeyStore.Entry`, což je rodičovská třída tříd `KeyStore.PrivateKeyEntry`, `KeyStore.SecretKeyEntry` a `KeyStore.TrustedCertificateEntry` reprezentující specifická data. Každý záznam (entry) v keystoru je identifikován svoji značkou (alias), pomocí níž lze také k datům přistupovat. List všech záznamů v daném keystore lze získat pomocí následujícího kódu:

```
aliases = ks.aliases();
```

Příslušný záznam lze získat pomocí kódu:

```
KeyStore.Entry entry = getEntry (String alias,
KeyStore.ProtectionParameter param)
```

Kde alias je výše zmíněná značka a param je zabezpečení tohoto záznamu. Každý záznam lze totiž dle uvážení uložit zabezpečeně i v rámci keystoru.

6 Návrh a implementace přístupového systému

V rámci bakalářské práce byl navržen autentizační systém, určený k fyzickému řízení přístupu. V rámci návrhu systému a jeho implementace se vycházelo ze získaných poznatků z předchozích kapitol z oblasti kryptografie a vývoje aplikací pro OS Android.

V dnešní době je časté, že uživatelé se v takovýchto systémech autentizují předmětem (většinou kartou). Myšlenkou aplikace je možnost uživatelů využít místo karet mobilní telefony. Případně lze uživatelům umožnit jak autentizaci pomocí karty, tak i pomocí telefonu, což umožní integraci vytvářeného systému do systémů stávajících. Tato varianta je velice výhodná i z toho důvodu, že ne každý uživatel vlastní zařízení s OS Android podporující NFC.

Implementaci přístupového systému lze rozdělit na dvě části. První částí je aplikace určená pro mobilní zařízení s OS Android. Druhou částí je aplikace spuštěná na straně terminálu. Schéma zapojení jednoho terminálu lze vidět na Obr. 17. Terminál s vytvořenou aplikací slouží k obsluze dat přijatých čtečkou, zajišťuje ověření uživatele, přidání nových uživatelů do systému a vytváření logů. Je spojen s databází, ve které jsou uložena všechna data o uživatelích. Do této databáze případné nové uživatele zapisuje. Aby nebylo třeba zasahovat do databáze příliš často, stahuje terminál ihned po spuštění celou databázi uživatelů. Logy v podobě textových souborů jsou ukládány na FTP server. Při použití více terminálů je databáze a FTP server sdílen.

Obr. 17 Schéma zapojení přístupového bodu přístupového systému

6.1 Klientská aplikace na zařízení s OS Android

Aplikace pracující na zařízeních s OS android byly vyvinuty dvě. První z nich demonstruje použití peer to peer módu. Výhodou této aplikace je možnost instalace na starší zařízení. Nejnižší verze je však Android 4.0 (API verze 14). Zařízení musí samozřejmě mít hardwarovou podporu NFC. Druhá aplikace demonstruje využití funkce emulátoru karet. Její výhodou je především uživatelská jednoduchost. Bohužel může být spuštěna pouze na zařízeních s OS Android verze 4.4 nebo novější, které navíc mají hardwarovou podporu NFC. Tato zařízení nejsou v současné době mezi běžnými uživateli hojně rozšířena.

Přestože obě aplikace využívají jiný druh zpráv (peer to peer aplikace NDEF a HCE APDU zprávy), některé prvky mají tyto aplikace společné. Každá vyslaná APDU zpráva (HCE) či NDEF záznam (peer to peer) má své ID, které určí, co se v příslušné zprávě přenáší. Výčet všech ID použitých v systému je uveden v Tab. 1. Obě aplikace pracují vždy v jednom ze čtyřech módů. Informace o druhu módu je uvedena ve zprávě s ID MODE. Možné hodnoty jsou patrné z Tab. 2. Jedná se o:

- **Pair mód** – slouží pro přenos uživatelských dat (Android ID, AES (*Advanced Encryption Standard*) klíč, RSA (*Rivest, Shamir, Adleman*) certifikát a heslo) aplikaci na straně terminálu. Jedná se tedy o fázi registrace uživatele do systému.
- **Autentizační módy** – slouží pro autentizaci uživatele. Jedná se o všechny módy kromě pair módu. Takové módy jsou tři: AES mód, RSA mód a SHA1 (*Secure Hash Algorithm*) mód.

Tab. 1 Výčet ID zpráv používaných v systému a jejich hodnoty

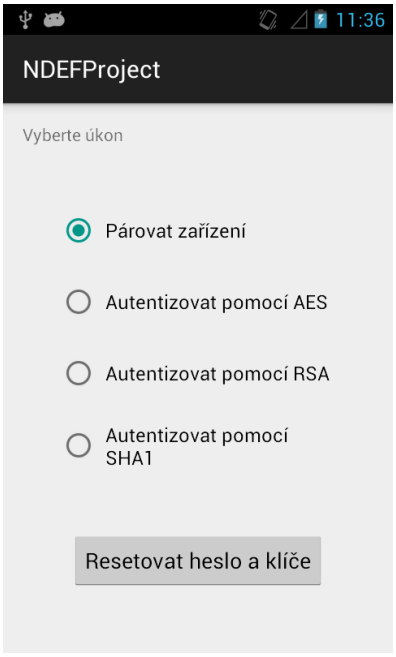
Název	Hodnota (hexadecimální)
MODE	0x00
PASSWORD	0x01
RSA_PUBLIC_KEY	0x02
RSA_PRIVATE_KEY	0x03
AES_KEY	0x04
ENCRYPTED_MSG_AES	0x05
ENCRYPTED_MSG_RSA	0x06
ENCRYPTED_MSG_SHA1	0x07
ANDROID_ID	0x08
CHALLENGE	0x09
TIME	0x10
SELL_APP	0x11
SEND_NEXT	0x12
NAME	0x13
RSA_CERTIFICATE	0x14

Tab. 2 Možné módy používané v rámci systému a hodnoty, které je identifikují (přenášeny jako tělo ve zprávě s ID MODE)

Název	Hodnota (hexadecimální)
AES mód	0x00
RSA mód	0x01
SHA1 mód	0x02
PAIR mód	0x03

6.1.1 Aplikace pro peer to peer mód

Předpokládané využití peer to peer módu v zařízeních s OS Android je například přenos kontaktů, fotek, SMS (Short Message Service) zpráv a dalších informací mezi dvěma mobilními zařízeními. Nepředpokládá se tedy, že by přenos měl být obousměrný. Při vývoji aplikace bylo tedy potřeba brát tento fakt v úvahu, což způsobilo i odlišný způsob autentizace ve srovnání s aplikací pracující jako emulátor karet. Uživatelské rozhraní aplikace lze vidět na Obr. 18.



Obr. 18 Uživatelské rozhraní peer to peer aplikace

Jelikož přenos probíhá jednosměrně, určuje mód aplikace uživatel. Aplikace na straně terminálu pouze přijímá vyslané zprávy. Formát zpráv je NDEF, což je pro peer to peer mód typické. V následujícím jsou popsány funkce jednotlivých módů přenosů, které uživatel vybere označením příslušného tlačítka.

Párování zařízení – pokud uživatel označí tuto možnost, aktivuje se pair mód. Používá se v případě, kdy chce svůj mobilní telefon registrovat do databáze známých zařízení, ovládané aplikací spuštěnou na straně terminálu. Pokud se uživatel pokusí autentizovat pomocí některých z ostatních módů, aniž by před tím použil párování, bude odmítnut.

Autentizovat pomocí AES, RSA a SHA1 – Pokud uživatel zvolí jeden z těchto módů, odešle aplikace autentizační zprávu. Aplikace obsluhující čtečku tuto zprávu zpracuje a vyhodnotí, zda má být uživatel autentizován či nikoliv.

Tlačítko Resetovat heslo a klíče – Z důvodu bezpečnosti může být vyžadováno pravidelné resetování hesel a klíčů. Po stisku tohoto tlačítka budou tyto úkoly vykonány a budou vytvořeny klíče nové. Je zřejmé, že uživatel nebude po takové změně schopen autentizace a bude se muset opět registrovat pomocí pair módu.

Implementace aplikace

Po přiložení aplikace ke čtečce se zobrazí výzva k provedení přenosu dat. Po dotyku obrazovky je přenos vykonán. V každém módu jsou samozřejmě odeslána jiná data. Vlastní NDEF zpráva sestává z několika NDEF záznamů viz Obr. 9. Ve všech módech je zasílán jako první záznam, který rozliší, o jaký mód komunikace se jedná (s ID MODE). Obsahuje jednu ze čtyř možností: PAIR_APP, AES, RSA a DIGEST_SHA. Každá možnost je representována vlastní bajtovou hodnotou. Další záznamy jsou závislé na módu aplikace.

V pair módu se vysílají zprávy s ID: ANDROID_ID, RSA_CERTIFICATE, AES_KEY a PASSWORD. Android ID je unikátní identifikátor každého zařízení s OS Android. Získá se voláním:

```
String android_id =
Settings.Secure.getString(context.getContentResolver(),
            Settings.Secure.ANDROID_ID);
```

Pomocí Android ID jsou uživatelé vzájemně odlišeni. Password je heslo zadané uživatelem při prvním spuštění aplikace a slouží k ověření uživatele při použití SHA1. RSA certifikát obsahuje veřejný klíč, pomocí něhož lze uživatele ověřit v módu RSA, obdobně je využit AES klíč v módu AES. Struktura zprávy je patrná z Obr. 19:

Obr. 19 Struktura zprávy přenášené v pair módu při použití peer to peer aplikace

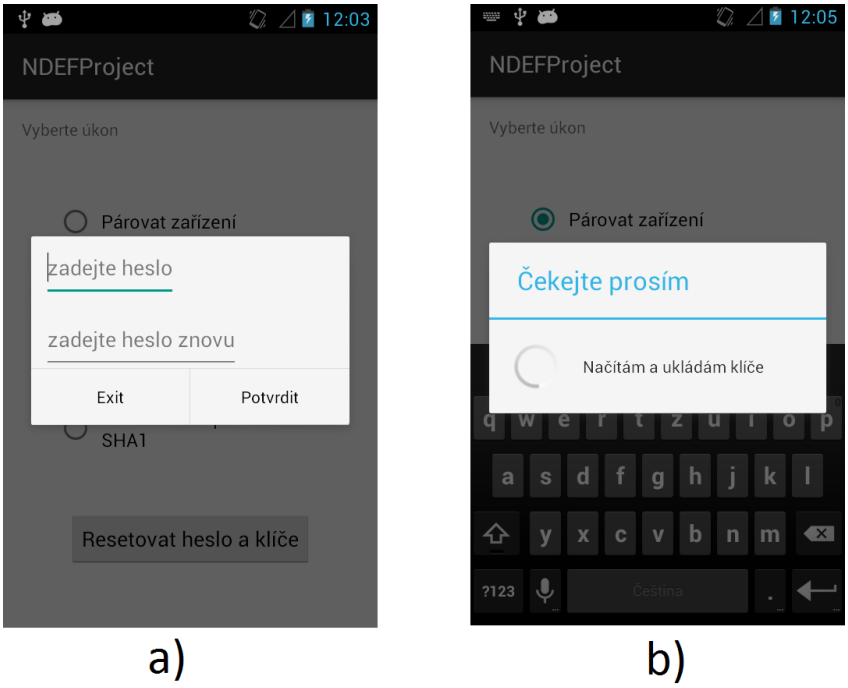
Autentizační módy jsou všechny módy kromě pair módu. S jejich pomocí se uživatel autentizuje. Jelikož je komunikace jednosměrná a současně je třeba zamezit reply útoku, je do každé zprávy přidána informace o současném čase. Na straně čtečky se potom ověřuje, zda není zpráva stará a pokud ano, nebude uživatel autorizován. Z tohoto důvodu je třeba, aby obě aplikace měly nastaven čas na stejnou hodnotu, nejlépe je použít čas ze sítě Internet. Časové okno, do kdy je zpráva považována za aktuální, je nastaveno na 2 sekundy. Kromě informace o čase nese zpráva záznam s hodnotou Android ID sloužící k identifikaci uživatele. Autentizační módy jsou celkem tři. AES mód, RSA mód a SHA1 mód. Jejich obecné schéma je shodné a je zobrazeno na Obr. 20.

- **Mód AES** – záznamy (současný čas a Android ID) jsou doplněny o zprávu zašifrovanou pomocí AES klíče, který byl odeslán v pair módu. Tato zpráva obsahuje bitové hodnoty Android ID a času.
- **Mód RSA** – v tomto módu se stejná zpráva (složená z času a Android ID) podepíše pomocí soukromého klíče, obsaženého v keystore. Tento podpis se spolu s informací o času a Android ID odesílá k terminálu. Logy jsou podobné logům AES, pouze je hodnota módu rovna hodnotě 0x01, což je identifikátor RSA módu.
- **Mód SHA1** – Zpráva, která byla dříve šifrována (případně podepisována), je doplněna o heslo zadané uživatelem a následně hašována. Záznam je doprovázen informací o času a Android ID.

Obr. 20 Struktura zprávy přenášené v některém z autentizačních módů při použití peer to peer aplikace

Zabezpečení klíčů v aplikaci

Po prvním spuštění aplikace je uživatel vyzván k zadání a potvrzení hesla. Toto heslo bude využito při zabezpečení dat v keystrore a také pro autentizaci při použití autentizačního módu SHA1. Po zadání hesla jsou vygenerovány klíče potřebné pro všechny druhy autentizace, viz Obr. 21. Jedná se o AES klíč délky 256 bitů, dále soukromý a veřejný klíč pro autentizaci pomocí RSA. Všechny tyto klíče jsou uloženy do keystoru. OS Android zaručuje, že data zde uložená bude velice těžké ze zařízení kopírovat na zařízení jiná. Keystore je navíc chráněn heslem vloženým uživatelem. Toto heslo je uloženo v shared preferences. Předáním parametru MODE_PRIVATE je zajištěno, že k těmto datum bude mít přístup pouze tato aplikace.



Obr. 21 Výzva uživatele na zadání hesla (a) a načítání klíčů po zadání hesla (b).

Testování aplikace

K ověření správné funkcionality dle teorie uvedené výše byly vytvořeny logy. Nejdříve jsou zobrazeny logy aplikace v pair módu a následně v módu AES. Logy ostatních módů nejsou zobrazeny, jelikož jsou velice podobné logům v AES módu.

- **Logy v pair módu:**

```
I/Main activity:  MODE

I/Main activity:  03

I/Main activity:  ANDROID_ID

I/Main activity:  3366313030656361666565396161643 7

I/Main activity:  AES_KEY

feeb6db679d7e3fa2117f8f2658fa5d0ea3bd113f180cfec728f28a5080770cd

I/Main activity:  RSA_CERTIFICATE

I/Main activity:
308202973082017fa003020102020101300d06092a864886f70d0101050500300f31
0d300b060355040313045465737430 1e170d3135303532303130303534315a170d31
353036303631303436313 05a300f310d300b060355040313045465737430820122300
0d06092a864886f70d01010105000382010f003082010a0282010100c310ee0b7f3c
abab47cc62c136de428d061e8abb2bb24f9e450066235d416262edfa4323a4fceb37
fa4a246eed34b5082079df097bc24772927d4a8b0e29de026fc35661069f585ef8e1
afd70c407517e541f1b57ff0c7ab7c14721231ed2239060e16b17543ddaa265f0850
037589558b023e7953fdd05ea0210890055454cac265d425746559313abbee1ab26a
2cf7d7747f3c545a16d43d9cf5482dc5e46233102a3d05f961f0d63e2945a36ca2a3
14bcfcd1b668f86ebe4b55dc26284437fff2b21658aa66f4565a3102b374883f8b50
c9bd9fa76554fff80b781852ccf707c2e19cd7d233ab00f8cbdcefed01bca46dce94
5418265ffcdad26f811d00b90203010001300d06092a864886f70d01010505000382
01010025c8fffb06dc47a6a980c3ccbeafa17d140f47f78f292279158fe15e21c766
9b1800fb6ece59d461aa9cdb27ef59f0393e14cd3fcdef8654a8070c5cdaf1b1d97b
2ed4321b8d283295b7101e38f652b5044955857118ec549837f0ff041f4ecf89f22f
f63368c29cd8dec5779e0b5fc6b89fec7ae17e3e92b917672dca592bab36473ba769
ed8acbf06476706e94589f1128275bf20d061e25b641cdf0162bbaa0f2f543e395d8
be738848fee102b3d1c616813ae5b0e1a4cb5e660357ce7653c1f5ba00e88c586000
27125ec5b33e171ff8bb3cda5d848a61c41b0780a5684962c5ae9805bbf26c8fba5d
c6b583418a21285d2e116ec5ac7c18c22533c43275

I/Main activity:  PASSWORD

I/Main activity:  616161
```

V logu je vždy uvedeno ID záznamu. Jedná se o jednu z hodnot zobrazených v Tab. 1. Pod tímto údajem jsou potom zobrazena přenášená data (tělo). Z logů je patrné, že přenášená data se shodují s teoretickou strukturou na Obr. 19 a také mód (tělo v záznamu s ID MODE) odpovídá hodnotě pair módu uvedené v Tab. 2.

- **Logy v AES módu:**

```
I/Main activity:  MODE
I/Main activity:  00
I/Main activity:  ANDROID_ID
I/Main activity:  3366313030656361666565396161643 7
I/Main activity:  TIME
I/Main activity:  0000014d711ed928
I/Main activity:  ENCRYPTED_MSG_AES
I/Main activity:
119c3fb5ef24146445abd4820e580f14a0728937a431096c9fa19d5ba9854cbe
```

Formát autentizačního módu využívající symetrickou blokovou šifru AES je stejný jako v případě ostatních autentizačních módů. I v tomto případě odpovídá struktura zprávy struktuře teoretické z Obr. 20. a data ve zprávě s hodnotou ID nastavenou MODE se shodují s Tab. 2.

6.1.2 Aplikace pro HCE

Aplikace je založena na technologii HCE a je v mnoha ohledech jiná než aplikace pracující v peer to peer módu. Prvním rozdílem je, že komunikace je obousměrná a to, že komunikaci zahajuje aplikace na straně terminálu. Místo NDEF zpráv jsou zde posílány APDU zprávy, které jsou typické pro komunikaci s čipovými kartami. V této aplikaci je využito stejných módů jako u peer to peer aplikace.

Specifikace APDU zpráv a odpovědí v aplikaci

Jelikož maximální délka APDU zprávy (nebo odpovědi) je 255 bajtů a zprávy zasílané aplikací (především RSA certifikát odesílaný v pair módu) jsou delší, bylo třeba specifikovat, zda zasílaná zpráva je či není celá. APDU zprávy posílané čtečkou jsou velice krátké a tuto délku nepřesahují. V případě APDU odpovědí je možné mimo dat specifikovat dva bajty informační viz Obr. 11. V prvním z nich se přenáší informace o tom, zda došlo či nedošlo k chybě. Zpráva, která zde obsahuje hodnotu 0x90, zpravidla značí, bezchybnou komunikaci. V případě této aplikace byla ovšem přidána hodnota 0x91. Tato hodnota značí také bezchybnou komunikaci, nicméně informuje protější stranu o skutečnosti, že zpráva je příliš velká na to, aby ji bylo možné odeslat v jedné APDU odpovědi. Pokud by zpráva byla tak velká, že by nestačily ani dvě APDU odpovědi, bude druhá odpověď opět označena hodnotou 0x91. Obdobně by se pokračovalo s třetí čtvrtou a dalšími APDU odpověďmi. Pokud tedy příjemce příjme zprávu s prvním informačním bajtem označeným 0x91, zaznamená si data zprávy a čeká na zprávy další. Dokud se jedná o stejný druh zpráv (s prvním informačním bajtem 0x91), sloučí data z přijaté zprávy k datům ze zpráv předešlých. Po přijetí první

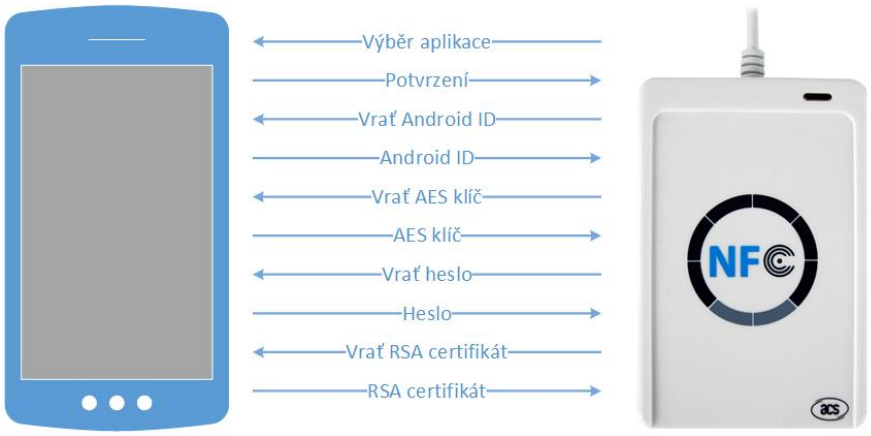
zprávy s prvním informačním bajtem nastaveným na 0x90 přidá data z této zprávy k datům nasbíraným z předešlého a zpráva je tímto přenesena celá. V druhém bajtu se potom přenáší informace o ID přenášené odpovědi. APDU zpráva má strukturu uvedenou na Obr. 10. Jednotlivé bajty jsou využity následujícím způsobem:

- **CLA** – hodnota 0x00 značí, že se jedná o zprávu specifikující aplikaci. Hodnota 0x80 potom značí klasickou zprávu.
- **INS** – parametr, který specifikuje mód komunikace. Hodnoty jsou stejné jako v peer to peer módu a jsou uvedené v Tab. 2.
- **P1** – tento parametr obsahuje ID zprávy. Hodnoty jsou opět stejné jako v peer to peer módu a jsou uvedené v Tab. 1.
- **P2** – parametr P2 není využit

Přenos dat v módu emulátoru karet

O tom, ve kterém módu bude komunikace probíhat, nerozhoduje zařízení s OS Android, ale aplikace obsluhující čtečku karet. Tato aplikace zasílá APDU zprávy a očekává APDU odpovědi. Zde budou popsány jednotlivé módy přenosu.

V pair módu aplikace obsluhující čtečku požaduje zaslání informací, které potřebuje pro pozdější autentizaci uživatele. Jedná se o stejné prvky jako v případě peer to peer (heslo, RSA certifikát a AES klíč). Zařízení na každou zprávu odpovídá příslušnou odpovědí, viz Obr. 22.



Obr. 22 Schéma přenosu dat v pair módu pro HCE aplikaci

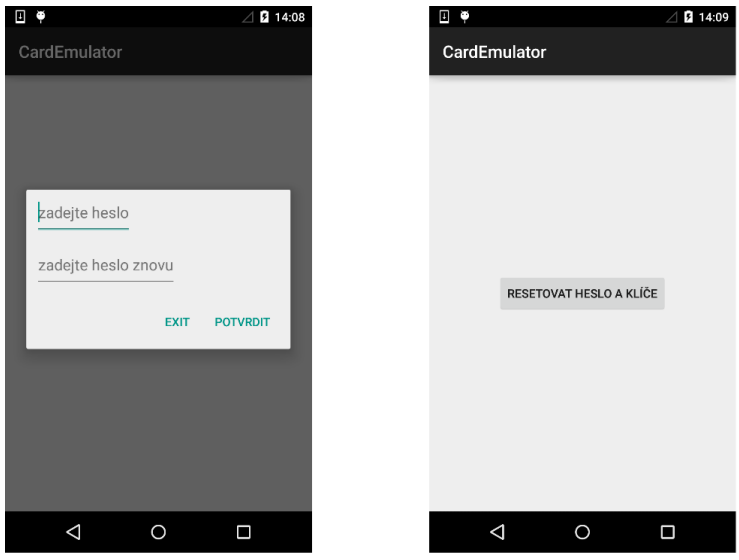
V autentizačních módech obdrží zařízení s OS Andoird unikátní zprávu s ID (v parametru P1) odpovídající prvku CHALLENGE. Podle módu komunikace (v parametru INS) rozhodne, jak s touto zprávou naložit. V případě módu AES přidá k výzvě své Android ID, zašifruje tajným klíčem a odesílá zpět. V případě RSA módu také přidá své Android ID, zprávu ovšem podepíše pomocí svého soukromého klíče vygenerovaného při párování. V módu SHA1 k odpovědi přidá Android ID a heslo. Takto vytvořený bitový tok hašuje pomocí algoritmu SHA1 a zprávu odešle zpět, viz Obr. 23.

Obr. 23 Schéma komunikace v autentizačním módu při použití HCE aplikace

Komunikace bude mít velice podobný charakter pro všechny módy autentizace, proto jsou zde uvedeny pouze logy vzniklé při použití SHA1.

Implementace HCE Aplikace

K plnění funkcionality slouží dvě komponenty. První z nich je aktivita s uživatelským rozhraním, která slouží k interakci s uživatelem. Při prvním spuštění vyžaduje od uživatele heslo, jehož funkčnost je stejná jako v případě peer to peer aplikace. Po zadání hesla také vygeneruje klíče a certifikát a bezpečně je uloží. Poté je zde možné resetovat heslo, klíče a certifikát pomocí příslušného tlačítka. Aktivitu lze vidět na Obr. 24.



a) b)

Obr. 24 Aktivita v módu emulátoru karty po prvním spuštění (a) a pro zadání hesla (b)

Druhou komponentou je služba dědící z třídy `HostApduService`. Tato služba odpovídá na zprávy zaslané čtečkou. A to voláním metody `processCommandApdu` po přijetí zprávy. Tato komponenta nemá uživatelské rozhraní (je spuštěná na pozadí) a umožňuje tak autentizaci bez interakce uživatele.

Testování aplikace

Podobně jako v případě peer to peer aplikace, byly vytvořeny logy demonstrující funkčnost aplikace dle teoretických předpokladů. Jedná se opět o dva záznamy logů, jeden v pair módu a druhý v SHA1 módu jako demonstrace autentizačních módů, kterou jsou velmi podobné.

- **Logy v pair módu:**

```
I/MyHostApduService: msg :00a4040006f06f6b61610100    cla 00 ins a4
p1 04 p2 00 lc 06 data f06f6b616101 le 00I/MyHostApduService: is
selected ApduI/MyHostApduService: response :    w1 90 w2 11 data
643638323133336632366233623764I/MyHostApduService: asked for :
ANDROID_ID I/MyHostApduService: msg :80030100
cla 80 ins 03 p1 01 p2 00 lc 00 data  le
I/MyHostApduService: response :900864363832313333663236623362376 4
    w1 90 w2 08 data 643638323133336632366233623764
I/MyHostApduService: asked for : AES_KEY
I/MyHostApduService: msg :80030400
    cla 80 ins 03 p1 04 p2 00 lc 00 data  le
I/key store helper: get AES key from keystore
I/MyHostApduService: response :w1 90 w2 04 data
710aef187009d978ed22e04a4aff41001d4c5530ca62f17756cbb3f09d9b8bf0
I/MyHostApduService: asked for : PASSWORD
I/MyHostApduService: msg :80030100
    cla 80 ins 03 p1 01 p2 00 lc 00 data  le
I/MyHostApduService: response :9001617177
    w1 90 w2 01 data 617177
I/MyHostApduService: asked for : RSA_CERTIFICATE
I/MyHostApduService: msg :80031400
    cla 80 ins 03 p1 14 p2 00 lc 00 data  le
I/MyHostApduService: response :
    w1 91 w2 14 data
308202973082017fa003020102020101300d06092a864886f70d0101050500300f31
0d300b060355040313045465737430 1e170d3135303532303134303933345a170d31
353036303631343530303 5a300f310d300b060355040313045465737430820122 30
0d06092a864886f70d01010105000382010f003082010a0282010100acd3f4d5de50
7a05351c2659642a5c5c7e590588c4551db47210b229b15953f80d02365fe0b0146f
897dd1d3666d89ece38cd25ae3a311ecd20668340a2cf96a407ec003473394354df5
f9e8cdc49bb15f15898dd782a74d0860251246adda4e2d29bfb22ab2964a32b444bb
c04665a4a03cc2bc2f8bfa34307864
I/MyHostApduService: response length 255
```

```
I/MyHostApduService: asked for : SEND_NEXT

I/MyHostApduService: msg :80031200

    cla 80 ins 03 p1 12 p2 00 lc 00 data  le

I/MyHostApduService: response :    w1 91 w2 14 data
b6ad542ab714ed859cffac002e4164b71ef1a6ecebd59ba1e15cf5477e03add81fa5
412d5d14d2c379e20c7ee660ae57738269414149baf2c22f8cf71d44ae757fe404d0
602a9e24dcaa0117430be91e3e304818b9d9905b12b617f196a6f78fa3eb3863bae0
cbe465534a2789039cddd095c631c0c899a877e88eef21ac6fc3224bb4498f020301
0001300d06092a864886f70d0101050500038201010038ea2bcdfab94dcccda79e8f
c7c816855ff57cdebc239ce0dbadd0d8ffcc0611687e896f34dbfb91db2adec350c7
7957bb790c3f46c88e79fceea189e7b7e6736699121eb691c478508858e1be32625e
21459c6249d7b8c58c2cee6ddd82fc

I/MyHostApduService: response length 255

I/MyHostApduService: asked for : SEND_NEXT

I/MyHostApduService: msg :80031200    cla 80 ins 03 p1 12 p2 00 lc
00 data  le05-20 17:08:04.935  14466-
14466/com.example.martin.cardemulator I/MyHostApduService: response
:    w1 90 w2 14 data
4d46202aa4009e5d8d30a54558cec73a64fa3efa26e514c1bdefc7d1c642bf173411
021e227ec452990070ce7dcf43e2f5b1a097c40a75b84a595b6f8f0f97b85ef147fa
31fdc33bca0e2cb653cbfdcdaa6f3df16a5e81c00ad975ea2aa5eb13eb4a15358aa7
b369e8fd0f5301fc16a63339da7a8d60690431d78046b4d2848fbf4acaa617442052
722c5f98d7a942ef81b0926457d2fd64a46da45be97e0a4216I/MyHostApduServic
e: response length 163
```

Z logů je zřejmá výše popsaná funkcionalita. V první zprávě označené žlutě se snaží čtečka vyhledat aplikaci v zařízení s OS Android. V dalších krocích jsou požadovány hodnoty Android ID, heslo, AES klíč a RSA certifikát. Z hlaviček odpovědí je patrné, že pro přenos RSA certifikátu byly využity celkem 3 zprávy, přičemž 2 z nich jsou označeny bajtem 0x91, o čem bylo pojednáno dříve. Tento fakt je znázorněn zeleně. Jsou zvýrazněny i délky zpráv. Je patrné, že ve zprávách s prvním bajtem nastaveným na hodnotu 0x91 mají maximální možnou délku 255 bajtů.

- **Logy v SHA1 módu:**

```
asked for : CHALLENGE

I/ApdMsg: je tam lc a data

I/MyHostApduService: msg :80020900080000014d72b045d8

    cla 80 ins 02 p1 09 p2 00 lc 08 data 0000014d72b045d8 le

I/Crypto alg helper: encrypting message
643638323133336632366233623764 0000014d72b045d8 617177

I/MyHostApduService: response
:9009c5cceaad36db3cbed7d9aa907a36e4080928de0a

    w1 90 w2 09 data c5cceaad36db3cbed7d9aa907a36e4080928de0a
```

Nejprve čtečka posílá zprávu s ID CHALLENGE (žlutě). Zařízení Android vytvoří zprávu získanou výše uvedeným způsobem. V této zprávě lze vidět část obsahující výzvu (zeleně). Tuto zprávu zašifruje (v tomto případě vypočte haš) a pošle zpět. Výsledná zpráva, která bude poslána zpět, je zvýrazněna tyrkysově.

6.1.3 Srovnání aplikací

Výhodou aplikace používající peer to peer mód je zpětná kompatibilita se staršími zařízeními Android. Nevýhodou je nižší bezpečnost. Jelikož je v tomto módu možný pouze jednosměrný přenos a je využito časové okno, kdy lze v období tohoto okna použít útok opakováním (Replu Attack) k dosažení neoprávněného přístupu. Pokud bude ovšem voleno okno dostatečně malé (2 sekundy), tak tento typ útoku není příliš pravděpodobný. Aplikace je relativně uživatelsky méně přívětivá. Uživatel musí při každé autentizaci zapínat autentizační aplikaci a posléze reagovat na výzvu čtečky dotykem. Může se navíc stát, že po výzvě na dotek uživatel nevědomky zařízení oddálí od čtečky, čímž dojde k přerušení komunikace mezi zařízením a čtečkou.

Naproti tomu aplikace pracující v módu emulátoru karet umožňuje komunikaci obousměrnou. Díky tomuto faktu lze provádět klasickou autentizaci výzva odpověď a zaručit, aby každá zpráva výzva byla unikátní. Tímto je možnost útoku opakováním znemožněna. Aplikace emulátoru karet je také uživatelsky přívětivější, jelikož uživatel nemusí ani odblokovat zařízení. Stačí jej pouze přiložit ke čtečce, čímž dojde k zahájení autentizačního procesu.

6.2 Aplikace na straně terminálů

Aplikace spuštěná na straně terminálu mimo jiné zpracovává data odeslaná zařízením Android nebo naopak data na takové zařízení posílá. Tuto aplikaci lze rozdělit na dvě části (nejedná se ovšem o dvě aplikace jako v případě aplikací pro OS Android). První část komunikuje se zařízením pracujícím v peer to peer módu a druhá pracuje se zařízením v módu emulátoru karet. Tyto části aplikace mají společné uživatelské rozhraní, zobrazené na Obr. 26. Mají také společnou databázi uživatelů a místností.

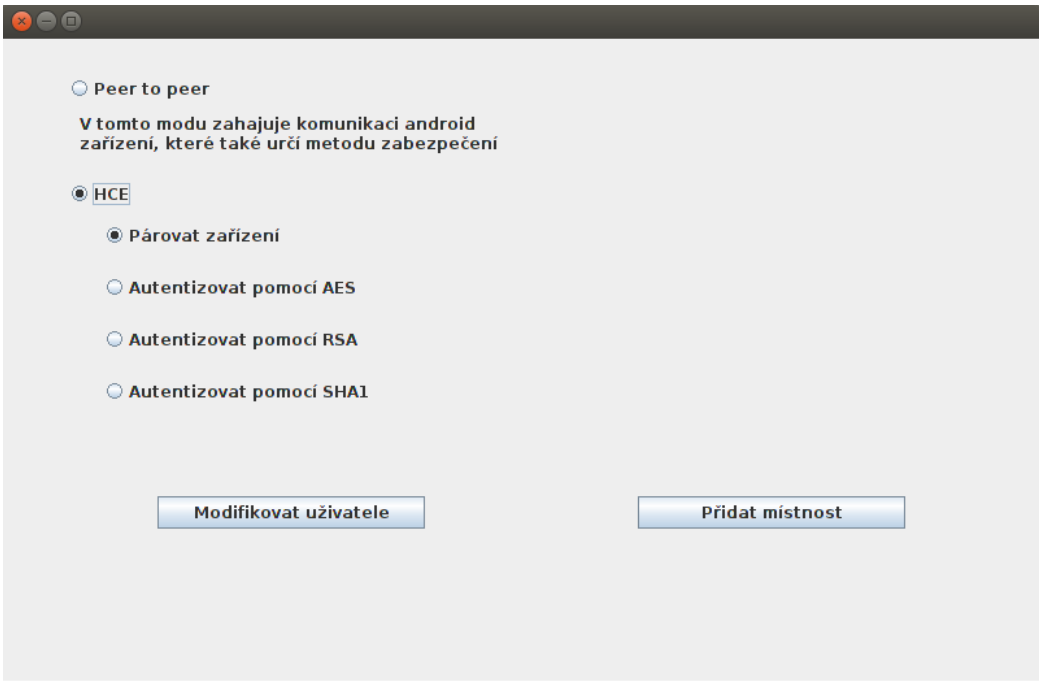
Databáze byla při implementaci spravována přes webové rozhraní pomocí softwaru phpMyAdmin, jenž je součástí programu XAMPP, který slouží k simulaci webového serveru na PC. Konktrétně byly v programu XAMPP k ukládání dat do databáze užity komponenty Apache serveru a MySQL (*Structured Query Language*). Tato databáze je chráněna uživatelským jménem a heslem, které jsou po uživateli vyžadovány při startu aplikace. Databáze obsahuje tři tabulky. První z nich obsahuje seznam uživatelů přidaných pomocí pair módu. Tato tabulka obsahuje pro každého uživatele informace o jeho jméně, zadaném při

párování, hesle, AES klíči a RSA certifikátu. Druhou tabulkou je tabulka místností, přidaných k danému uživateli, čili takových, do kterých má uživatel přístup. Tato tabulka obsahuje ID uživatele a ID místnosti. Obě tyto hodnoty jsou unikátní. Poslední tabulkou je pak tabulka místností. Tato tabulka obsahuje ID místnosti a její název. Obě hodnoty musí být unikátní. ID místnosti je automaticky inkrementováno, což zaručí jeho unikátnost. Struktura jednotlivých tabulek je zobrazena na Obr. 25. Data o všech uživatelích a jejich právech jsou stažena při startu aplikace, aby se předešlo častému přístupu do databáze. V průběhu činnosti aplikace se seznam uživatelů udržuje aktuální a do databáze se nahrávají pouze případné změny.

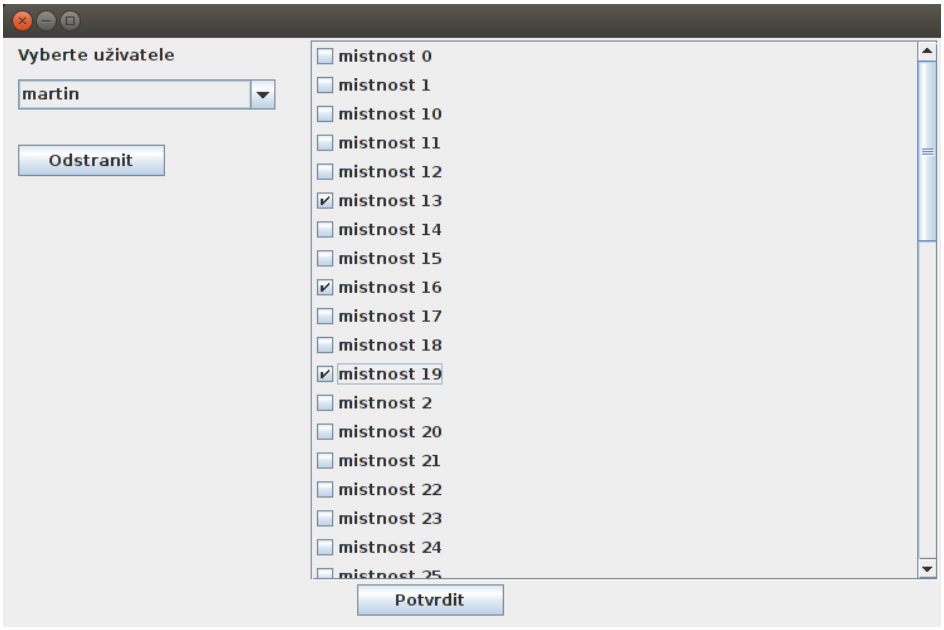
Aplikace zaznamenává i výsledky autentizace či registrace uživatelů. Tato data ve formě textového souboru jsou uložena na FTP server, který je opět spuštěn pomocí programu XAMPP. V tomto případě se ovšem jedná o komponentu FileZila. I zde je třeba, aby se uživatel autentizoval jménem a heslem.

Obr. 25 Struktura tabulek databáze uživatelů, spojovací tabulky a místností

Mimo možnosti volby aplikace nabízí základní GUI (*Graphical User Interface*) další dvě možnosti. První z nich je modifikace uživatele. Po stisknutí tohoto tlačítka lze uživatele mazat nebo přidávat či odebírat práva uživatelů k jednotlivým místnostem viz Obr. 27. Druhou možností je přidat novou místnost. Po stisku toho tlačítka je po uživateli vyžádáno jméno nové místnosti. Místnost je potom uložena do příslušné tabulky v databázi.



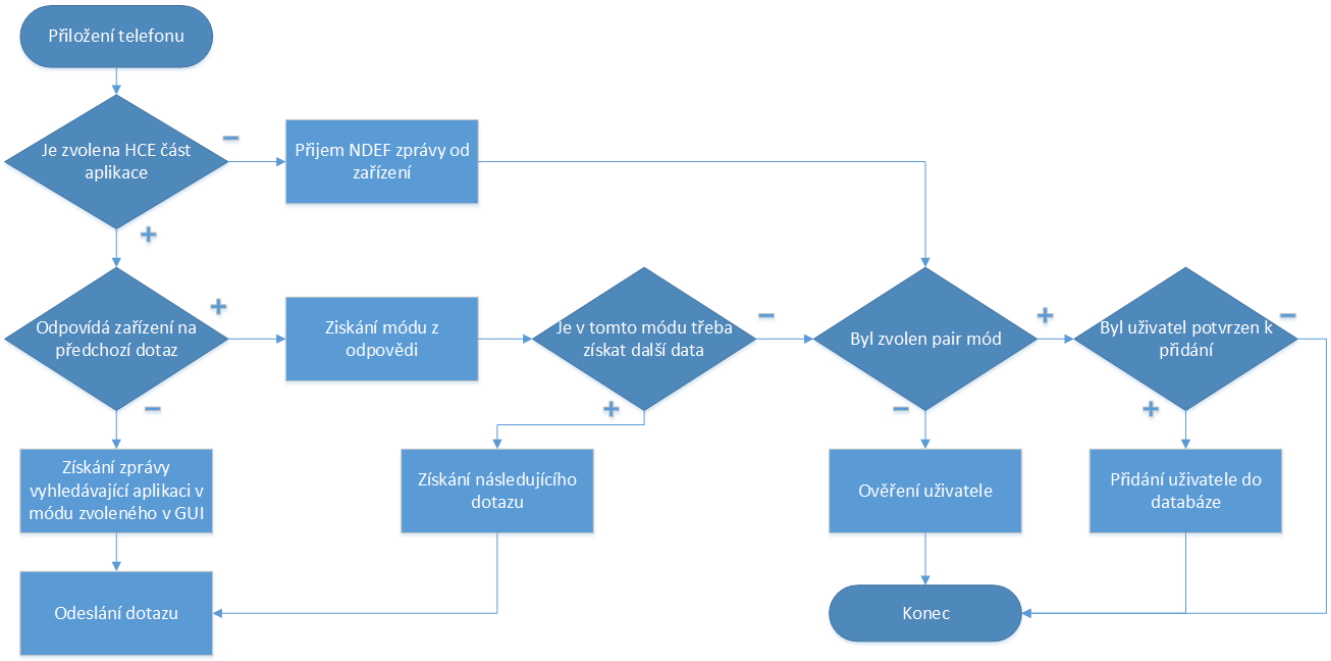
Obr. 26 Základní uživatelské rozhraní



Obr. 27 Uživatelské rozhraní pro modifikaci uživatelů

6.2.1 Popis jednotlivých módů

Jak již bylo zmíněno, aplikace uchovává aktuální seznam uživatelů, který stahuje při svém spuštění. Při autentizaci uživatele tedy není spojení s databází vůbec zapotřebí. Data zasílaná mezi aplikací na straně terminálu a aplikací na zařízení s OS Android, byla popsána v kapitole 6.1.1 a 6.1.2. Vývojový diagram komunikace aplikace je zobrazen na Obr. 28.



Obr. 28 Vývojový diagram komunikace aplikace na straně terminálu

V pair módu se vyhledá uživatel pomocí ID, které odeslal. Pokud se v databázi již takový uživatel nachází, reaguje aplikace chybovým hlášením. V opačném případě požádá obsluhu terminálu o zadání jména nového uživatele a po jeho zadání je uživatel uložen do seznamu uživatelů i do databáze a informuje o této skutečnosti uživatele. Výsledné zprávy lze vidět na Obr. 29.

Obr. 29 Pokus o autentizaci uživatele, který není v databázi (a) a přidávání nového uživatele do databáze (b)

V ostatních módech ověřuje aplikace uživatele pomocí dat uložených v databázi. V první fázi je třeba ověřit, zda jsou v databázi o uživateli uložena nějaká data. Pokud tomu tak není, je uživatel informován chybovým hlášením. V opačném případě se načtou data potřebná pro autentizaci ze seznamu uživatelů. Nejprve se ověří, zda má uživatel přístup k místnosti, do které žádá přístup, viz Obr. 30 a). Pokud ne, je opět zobrazeno chybové hlášení. Pokud je

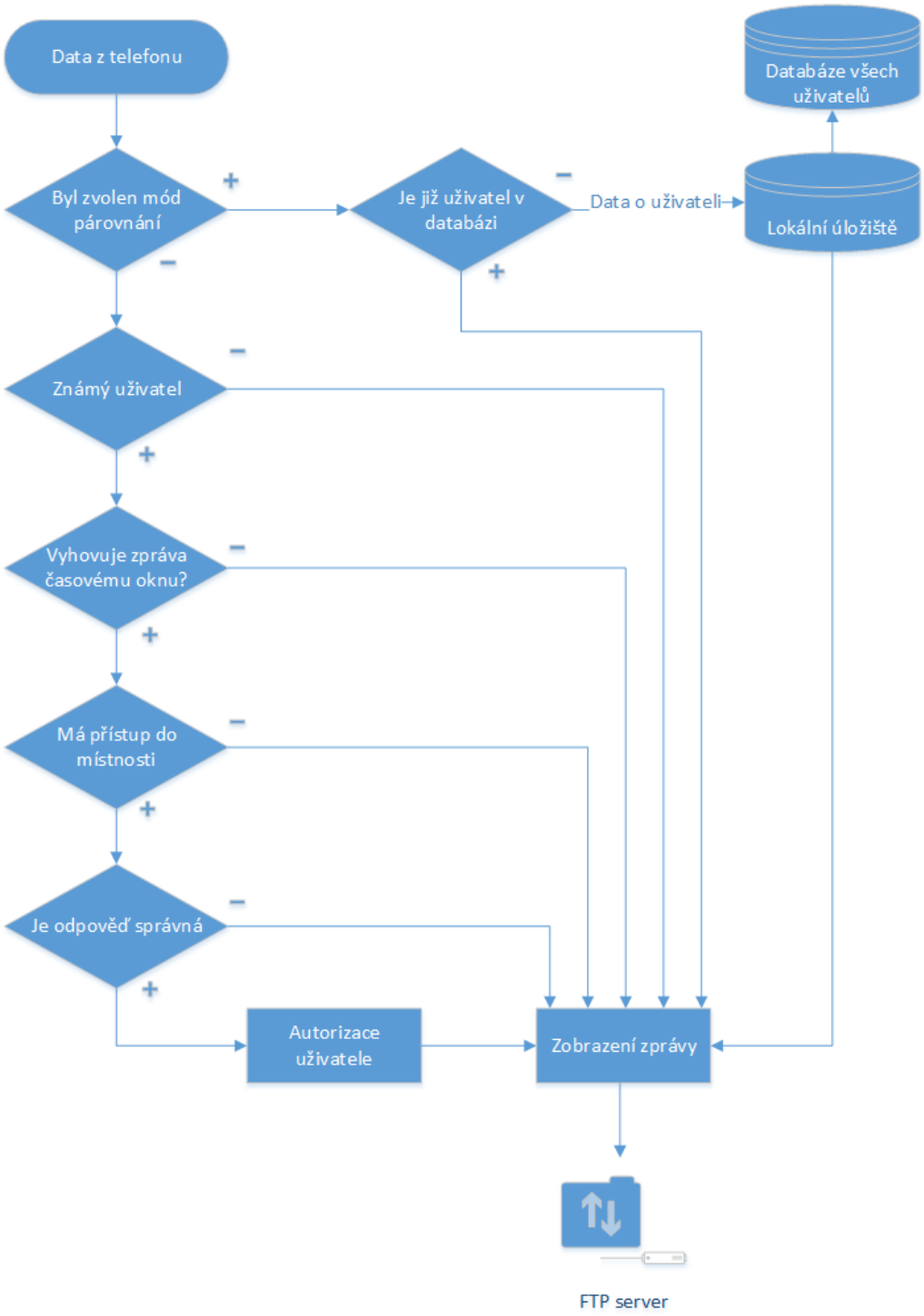
vše v pořádku, ověří se zpráva přijatá ze zařízení Android s ID challenge. V případě AES módu se dešifruje zpráva pomocí klíče uloženého v databázi a ověřuje se, zda jsou výsledná data taková, jaká byla očekávána. V případě módu RSA se pomocí veřejného klíče uloženého v certifikátu ověří, zda byla přijatá zpráva podepsána správným zařízením. V módu SHA1 se vytvoří haš z dat výzvy, Android ID a hesla a porovná se s haší přijatou od zařízení s OS Android. Pokud jsou tyto zprávy stejné, je uživatel autentizován, jinak nikoliv. V obou případech je uživatel informován o výsledku procesu ověření.

a) b)

Obr. 30 Pokus o neoprávněný vstup uživatele do místnosti (a) a úspěšná autorizace (b)

6.2.2 Část aplikace komunikující se zařízením v peer to peer módu

Při vývoji této aplikace bylo použito API rozhraní java-android-beam-api dostupné z [19]. Pro správnou funkčnost bylo třeba vyvíjet aplikaci v OS Linux (byla zvolena distribuce Ubuntu 14.04.2), a to 64 bitovou verzi tohoto operačního systému. Tuto aplikaci lze spustit označením příslušného tlačítka v základním uživatelském rozhraní. Mód komunikace zde vybírá zařízení s OS Android. Spojení je pouze jednosměrné a tato aplikace tedy data pouze přijímá a zpracovává. Dle módu určí, zda má data uložit do SQL databáze (pair mód), či se pokusit v této databázi vyhledat příslušného uživatele identifikovaného svým Android ID. Na Obr. 31 lze vidět vývojový diagram práce této aplikace.

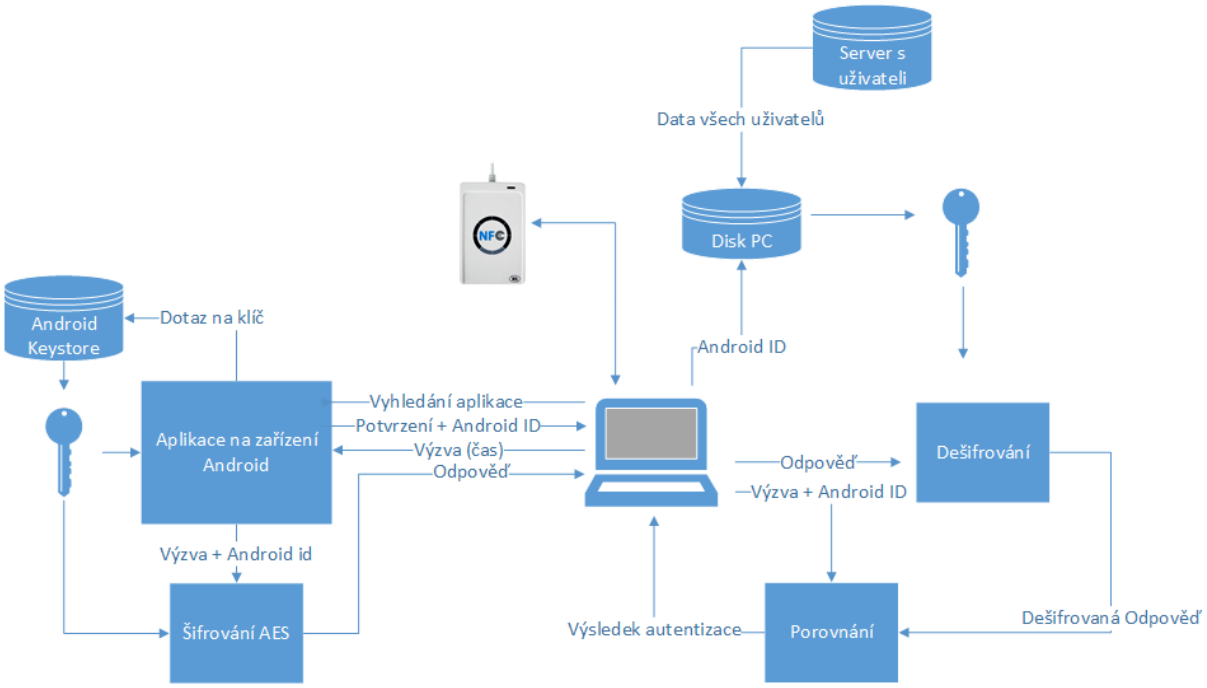


Obr. 31 Vývojový diagram aplikace pracující v peer to peer módu

6.2.3 Aplikace komunikující se zařízeními v módu emulátoru karet

Při vývoji této aplikace byla využita knihovna javax.smartcardio, dostupná z [10]. Stejná knihovna bývá použita i při vývoji aplikace pracujícími s kartami. K jejímu výběru dojde

pomocí označení tlačítka „HCE“. V tomto módu zahajuje komunikaci právě tato aplikace a specifikuje také mód komunikace, pomocí dalších tlačítek zobrazených pro stisku na tlačítko „HCE“. Vývojový diagram je velice podobný diagramu z části peer to peer na Obr. 31. Rozdílem je pouze to, že data jsou získána pomocí více zpráv, jelikož komunikace je obousměrná, viz Obr. 22. V tomto módu také není kontrolováno časové okno. Celý proces autentizace pomocí aplikace HCE v módu AES je znázorněna na Obr. 32.



Obr. 32 Zobrazení procesu autentizace AES při použití HCE aplikace

6.2.4 Zaznamenávání událostí

Všechny činnosti na straně terminálu jsou zaznamenány ve formě logů a jsou uloženy na FTP server. Soubory jsou ukládány pod názvem odvozených z názvů místností ke kterým se vztahují. Jelikož jména místností jsou unikátní, nemůže dojít k vzájemnému přepisování souborů. Ukázka souboru s logy:

```
2015/05/21 13:06:25  Místnost mistnost 14 Registrace uživatele
zamítnuta
2015/05/21 13:06:52  Místnost mistnost 14 Uživatel martin úspěšně
přidán do databáze
2015/05/21 13:07:32  Místnost mistnost 14 Uživatel martin nemá práva
pro vstup do místnosti !
2015/05/21 13:08:57  Místnost mistnost 14 Uživatel martin
autorizován pomocí SHA
2015/05/21 13:09:22  Místnost mistnost 14 Toto zařízení je již v
databázi registrováno !
```

```
2015/05/21 13:10:15  Místnost mistnost 14 Uživatel martin
autorizován pomocí SHA

2015/05/21 13:13:14  Místnost mistnost 14 Uživatel martin nebyl
autorizován
```

Z logů je zřejmé, že poprvé byl uživatel zamítnut s žádostí o párování (1. řádek). Potom byl již přidaný pod jménem martin. Při jeho první autentizaci ovšem neměl dostatečná práva k přístupu do místnosti. Po přidání práv byl již bez problémů autorizován a použil metodu SHA1. V 6. řádku byl uskutečněn pokus přidat uživatele, který již v databázi je. V 7. řádku byl uživatel opět úspěšně autorizován. Podle posledního řádku lze usoudit, že uživatel martin nebyl autorizován. Tento stav je způsoben například resetováním klíčů a hesel v zařízení s OS Android.

Závěr

V bakalářské práci byly popsány možnosti využití mobilního telefonu jako autentizačního předmětu. V kapitole 1 byly popsány základy kryptografie. V kapitole 2 potom druhy autentizace. Další dvě kapitoly se věnovaly moderním technologiím, které mohou být využity právě v oblasti fyzického řízení přístupu. Mezi tyto technologie patří NFC a BLE. Bylo naznačeno, jak vyvíjet aplikace využívající tyto technologie. Každá z technologií má oproti druhé výhody, které byly v práci zmíněny. V páté kapitole byl popsán způsob, jakým lze bezpečně uložit data v zařízeních s OS Android.

Na základě teoretických poznatků byly vytvořeny tři aplikace, které demonstrují použití NFC k účelům autentizace. Tyto tři aplikace se společně podílejí na funkčnosti přístupového systému. Dvě z nich jsou nainstalovány na zařízení s OS Android a jedna z nich potom obsluhuje čtečku karet a řídí komunikaci s databází a FTP serverem. Pomocí unikátního čísla AID, operační systém Android určí správnou aplikaci pro komunikaci, je tedy zajištěno, že vytvořená aplikace bude reagovat pouze s takovou aplikací na straně terminálu, která bude tento identifikátor sdílet. Tím je dosaženo toho, že uživatel nemusí při autentizaci volit z případných mnoha aplikací nainstalovaných v jeho mobilním telefonu, sloužících k podobnému účelu.

Aplikace, které jsou kompatibilní se systémem Android, mají některé odlišnosti. Liší se v módu komunikace. První z nich využívá peer to peer módu, čímž umožňuje použití i uživatelům se staršími zařízeními s OS Android (nejstarší však Android 4.0). Tato aplikace je uživatelsky méně přívětivá a méně bezpečná, jelikož lze provést reply attack. Druhá z nich využívá velice perspektivní mód emulátoru karet, kdy zařízení s OS Android komunikuje pomocí zpráv stejného formátu jako skutečné karty. Aplikace je také uživatelsky přívětivější a reply attack je zde vyloučen. Její nevýhodou je použití na poměrně úzkém spektru zařízení (vyžaduje verzi OS Android 4.4 nebo novější a podporu NFC).

Výhodou implementovaného systému je, že uživatel musí mít k jeho použití u sebe pouze mobilní telefon, což je v dnešní době v podstatě nutností. Navíc jeden telefon může vystupovat jako více karet pro různé čtečky. Výhodou mobilního telefonu ve srovnání se standardní kartou je vyšší výpočetní výkon. Jelikož mobilní telefony jsou výkonné, bylo možné využít různých kryptografických operací pro zabezpečení. Konkrétně byly použity algoritmy SHA1, AES a RSA.

Autentizační protokol pracující s technologií BLE nemohl být realizován, jelikož dostupný telefon Nexus 5 nemá hardwarovou podporu pro BLE v módu periferie.

Literatura

[1] ALLEN, Grant. *Android 4: průvodce programováním mobilních aplikací*. 1. vyd. Překlad Jakub Mužík. Brno: Computer Press, 2013, 656 s. ISBN 978-80-251-3782-6.

[2] *Android developers* [online]. [cit. 2014-12-13]. Dostupné z: http://developer.android.com/index.html

[3] *Bluetooth* [online]. [cit. 2014-12-13]. Dostupné z: https://developer.bluetooth.org/Pages/default.aspx

[4] Bluetooth Versions Walkthrough, and Bluetooth 4.0 Low Energy Development Resource. *Cnx-software* [online]. 2013 [cit. 2014-12-14]. Dostupné z: http://www.cnx-software.com/2013/06/05/bluetooth-versions-walkthrough-and-bluetooth-4-0-low-energy-development-resources/

[5] BURDA, Karel a Ivo STRAŠIL. *Zabezpečovací systémy*. Brno, 2011.

[6] BURDA, Karel. *Bezpečnost informačních systémů*. Brno, 2005.

[7] DZURENDA, Petr. *KRYPTOGRAFIE V INFORMATICE (MKRI): Programovatelné čipové karty .NET (autentizace)*. Brno, 2014.

[8] HAJNÝ, Jan. *Přednáška předmětu BZKR - Základy kryptografie 2*.

[9] Chytré telefony jako náhrada čipových karet. *Systemonline* [online]. [cit. 2014-12-14]. Dostupné z: http://www.systemonline.cz/it-security/chytre-telefony-jako-nahrada-cipovych-karet.htm

[10] *Javax.smartcardio* [online]. 2015 [cit. 2015-05-20]. Dostupné z: https://docs.oracle.com/javase/8/docs/jre/api/security/smartcardio/spec/javax/smartcardio/package-summary.html

[11] KOEGH, James. *Java bez předchozích znalostí: průvodce pro samouky*. Vyd. 1. Brno: Computer Press, 2005, 274 s. ISBN 80-251-0839-2.

[12] MENEZES, Alfred J. *Handbook of applied cryptography*. Vyd. 1. Boca Raton: CRC Press, 1997, 780 s. ISBN 08-493-8523-7.

[13] MURPHY, Mark L. *Android 2: průvodce programováním mobilních aplikací*. Vyd. 1. Brno: Computer Press, 2011, 375 s. ISBN 978-80-251-3194-7.

[14] NFC technologie - odborný pohled na funkčnost a využití v praxi. *Mobilizujeme* [online]. 2011 [cit. 2014-12-13]. Dostupné z: http://mobilizujeme.cz/clanky/nfc-technologie-odborny-pohled-na-funkcnost-a-vyuziti-v-praxi/

[15] OCHODKOVÁ, Eliška. *Matematické základy kryptografických algoritmů*. Ostrava, 2011.

[16] SMITH, Dave. Bluetooth Smart for Android. *Doubleencore* [online]. [cit. 2014-12-13]. Dostupné z: http://www.doubleencore.com/2013/12/bluetooth-smart-for-android/

[17] STALLINGS, William. *Cryptography and network security: principles and practice*. Seventh

edition. xix, 731 pages. ISBN 01-333-5469-5.

[18] TRČÁLEK, Antonín. Stačí přiložit: NFC a jeho využití v praxi. *Mobilmania* [online]. 2013 [cit. 2014-12-13]. Dostupné z: http://www.mobilmania.cz/clanky/staci-prilozit-nfc-a-jeho-vyuziti-v-praxi/sc-3-a-1325034/default.aspx

[19] Uses of Package de.estudent.accesscontrol.nfc.ndef. *Java-android-beam-api* [online]. 2012 [cit. 2015-05-20]. Dostupné z: https://java-android-beam-api.googlecode.com/svn/trunk/site/apidocs/de/estudent/accesscontrol/nfc/ndef/package-use.html

Použité zkratky

AES	Advanced Encryption Standard
AID	Application IDentifier
APDU	Application Protocol Data Unit
API	Application Programming Interface
BKS	Bouncy Castle Keystore
BLE	Bluetooth Low Energy
CLA	Class of Instruction
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DES	Data Encryption Standard
EPC	Electronic Product Code
FAR	False Acceptance Rate
FRR	False Rejection Rate
FTP	File Transfer Protocol
GATT	General Agreement on Tariffs and Trade
GUI	Graphical User Interface
HCE	Host Card Emulation
I/O	Input/Output
ID	IDentifier
IEC	International Electrotechnical Commission
INS	Instruction
JCEKS	Java Cryptography Extension Keystore
JKS	Java KeyStore
LC	Lenght of the Command data
LE	Length of the Expected response
MIME	Multipurpose Internet Mail Extensions
NDEF	NFC Data Exchange Format

NFC	Near Field Communication
PKCS12	Public-Key Cryptography Standards 12 Keystore
RFID	Radio Frequency IDentification
RSA	Rivest, Shamir, Adleman
SHA	Secure Hash Algorithm
SMS	Short Message Service
SQL	Structured Query Language
TELNET	TELecommunication NETwork
TNF	Type Name Field
URI	Uniform Resource Identifier
USB	Universal Serial Bus
UUID	Universally Unique IDentifier
XML	eXtensible Markup Language

Příloha

Příloha obsahuje následující:

NDEFProjekt2/	zdrojové kódy peer to peer aplikace pro zařízení s OS Android
NFCCardEmulator/	zdrojové kódy HCE aplikace pro zařízení s OS Android
Terminal app/	zdrojové kódy aplikace na straně terminálu
lib/	použité JAR soubory v aplikaci

Je také potřeba importovat knihovnu rt.jar, která z důvodu velikosti není umístěna v přiložených souborech. Tuto knihovnu je možné stáhnout z https://code.google.com/p/sand-a-java-ide-for-android/downloads/detail?name=rt.jar&can=2&q= nebo nalézt na přiloženém DVD.