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

BEZPEČNOSTNÍ APLIKACE PRO ANDROID

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

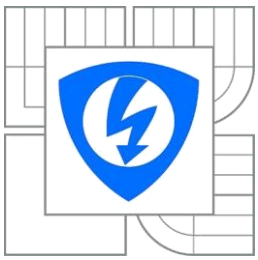
AUTOR PRÁCE — MICHAL CELENG

AUTHOR

VEDOUCÍ PRÁCE — Ing. JAN HAJNÝ, Ph.D.

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student: Michal Celeng ***ID:*** 146798
Ročník: 3 ***Akademický rok:*** 2013/2014

NÁZEV TÉMATU:

Bezpečnostní aplikace pro Android

POKYNY PRO VYPRACOVÁNÍ:

Nastudujte vývojové prostředí pro OS Android. Analyzujte možnosti prostředí pro výpočet operací s velkými čísly a modulární aritmetiku. Implementujte jednoduchý program pro měření časové náročnosti výpočtů s velkými čísly pro různé délky operandů a modula. Implementujte zadaný autentizační protokol pro zadané délky modula. Implementujte klientskou a ověřovací aplikaci pro zadaný protokol.

DOPORUČENÁ LITERATURA:

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

[2] HAJNÝ, J.; MALINA, L. Unlinkable Attribute-Based Credentials with Practical Revocation on Smart-Cards. In Smart Card Research and Advanced Applications. Lecture Notes in Computer Science. LNCS. Berlin: Springer- Verlag, 2013. s. 62-76. ISBN: 978-3-642-37287- 2. ISSN: 0302- 9743.

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

Termín zadání: 10.2.2014 ***Termín odevzdání:*** 4.6.2014

Vedoucí práce: Ing. Jan Hajný, Ph.D.
Konzultanti bakalářské práce:

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

ANOTACE

Práca sa zaoberá problematikou operácií s veľkými číslami v prostredí operačného systému Android, najmä ich časovou náročnosťou a implementáciou protokolu HM12 vytvoreného pre atribútovú autentizáciu. Výstupom vytvorenej aplikácie bude vygenerovaný QR kód s potrebnými informáciami pre overenie užívateľa. Predpokladom je zachovanie jeho anonymity a ďalších vlastností, ktoré sú v modernej kryptografii očakávané.

Klíčová slova : Android, trieda BigInteger, časová náročnosť, protokol HM12, modulárna aritmetika, QR kódy, klientská aplikácia

ABSTRACT

The work deals with the issue of operations with large numbers in the enviroment of the Android operating system, mainly their time consumption and the implementation of protocol HM12 created for attribute authentication. The output of created application is a QR code generated with the necessary information for the user authentication. A premise is to preserve user's anonymity and other features expected in a modern cryptography.

Keywords : Android, BigInteger class, time consumption, protocol HM12, Modular arithmetic, QR codes, client application

CELENG, M. *Bezpečnostní aplikace pro Android.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2014. 39 s. Vedoucí bakalářské práce Ing. Jan Hajný, Ph.D..

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma „Bezpečnostní aplikace pro Android“ jsem vypracoval(-a) 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 této bakalářské práce jsem neporušil(-a) autorská práva třetích osob, zejména jsem nezasáhl(-a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(-a) následku porušení ustanovení § 11 a následujících autorského zákona c. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonu (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4
Trestního zákoníku č. 40/2009 Sb.

V Brně dne 4. 6. 2014 .....

podpis autora

Poděkování

Ďakujem svojmu vedúcemu bakalárskej práce Ing. Janu Hajnému, Ph.D.za jeho metodickú pomoc, odborné vedenie, cenné rady a ústretovú komunikáciu počas celého trvania bakalárskej práce.

V Brně dne 4. 6. 2014 .....

podpis autora

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

Úvod

Digitálny svet je v dnešnej dobe pevne prepojený s bežným životom. Preto veľmi rýchlo rastie potreba chrániť svoje dáta, ktoré by mohli byť jednoducho zneužiteľné čoraz sofistikovanejším organizovaným zločinom v oblasti informačných technológií. Ďalšími frekventovanými termínmi a často diskutovanými otázkami sú anonymita vo virtuálnom prostredí, sledovanie bezpečnostnými agentúrami, zabezpečenie súkromia, zber obrovského množstva dát o jednotlivcoch a to či sa môže obyčajný človek uchrániť pred takýmto neželaným zberom informácií. Neopomenuteľným je aj fakt, že ľudia nie sú nadšení, keď je im znížený komfort alebo sa predlžuje doba, ktorú musia na výkon istej operácie alebo transakcie čakať.

Logickým dôsledkom je potreba nájsť prístroj, ktorý má dostatočné vybavenie na výpočtovo náročné operácie, je dostupný pre široké spektrum užívateľov, je maximálne mobilný a jeho každodenná prítomnosť je pre človeka takmer nevyhnutná. Pri opomenutí kariet a preukazov všetkého druhu sú jednoznačnou odpoveďou na tieto požiadavky dnešné inteligentné telefóny.

Táto práca, zaoberajúca sa vývojom pre operačný systém Android, ktorý je momentálne najrozšírenejším a najrýchlejšie sa rozvíjajúcim na celom svete, má za úlohu zmapovať možnosti použitia operácií s veľkými číslami v modulárnej aritmetike, ktorými sa zaoberá trieda BigInteger a je veľmi dôležitá pre implementáciu kryptografických schém do praxe. Súčasťou tejto časti práce je vytvorenie funkčného programu merajúceho časovú náročnosť jednotlivých funkcií a operácií v modulárnej aritmetike.

V ďalšej fáze práce sa zoznámime s konkrétnym protokolom, ktorý bol navrhnutý na Fakulte elektrotechniky a komunikačních technológií VUT v Brně. Jedná sa o protokol založený na vlastníctve osobných atribútov. Tento protokol bol zatiaľ prístupný iba pre čipové karty, súčasťou tejto práce je implementácia klientskej aplikácie pre operačný systém Android. Výstup aplikácie bude realizovaný pomocou QR kódu, ktorý je v súčasnosti veľmi často používaný a je na ňom možné zobraziť množstvo informácií.

Záverečnou fázou bude interpretácia zmeranej časovej náročnosti funkcií a implementovaného protokolu pre rôzne dĺžky modula, ich grafická závislosť a zdôvodnenie, či sú inteligentné telefóny s operačným systémom Android vhodné pre tieto aplikácie a použiteľné v praxi.

1 Operačný systém Android

Android je momentálne najpoužívanejšia platforma s otvoreným kódom pre mobilné zariadenia, ktorými sú dnes prevažne inteligentné telefóny, tablety a navigácie, najnovšie aj GoogleTV.

1.1 História

Firma na vývoj Androidu vznikla v októbri 2003 v Kalifornii pod názvom Android, Inc. V auguste 2005 odkúpila zakladateľskú firmu spoločnosť Google Inc. Tu sa začala vyvíjať platforma na linuxovom jadre. V novembri 2007 sa vytvorilo zoskupenie Open Handset Alliance, ktoré zastrešovalo spoločnosti ako Google, Samsung, HTC, Intel, NVidia, Motorola, LG alebo Texas Instruments. Toto zoskupenie kládlo veľký dôraz na vývojárov a po jednom týždni vydalo prvý vývojový program pre softvér pod voľnou licenciou. Prvé telefóny s operačným systémom Android sa dostali na americký trh v októbri 2008. Tento extrémne rýchly rozvoj charakterizuje celé pôsobenie Androidu vo svete technológii [1].

1.2 Architektúra

Architektúra Androidu je rozdelená na 5 vrstiev [2].

1. vrstva – Jadro operačného systému

Najnižšou vrstvou je jadro operačného systému, ktoré tvorí spojenie medzi hardvérom a softvérom, ktorý sa nachádza v štyroch vyšších vrstvách. Jadro funguje na báze Linuxu verzie 2.6, dôvodom pre jeho použitie je jednoduchá kompilovateľnosť na rôznych zariadeniach, ktorá je podmienkou pre jednoduchú distribúciu do rôznych zariadení. Android využíva funkcie jadra ako zabudované ovládače, správu pamäte, sietí a procesov alebo paralelný beh aplikácii, ktoré fungujú ako samostatné procesy s prioritou stanovenou systémom. Tieto vlastnosti sú základom pre funkčnosť a zabezpečenie systému.

2. vrstva – Knižnice

Knižnice sú písané v kóde jazykov C alebo C++ a sú využívané rôznymi časťami systému. Tie sú dostupné aj pre vývojárov vďaka 4. vrstve, ktorou je Application Framework. Implementované sú knižnice pre prácu s médiami (fotografie, obrázky, zvuk, video), databázami, 3D grafikou, webovým rozhraním a iné.

3. vrstva – Android Runtime

Náplňou práce tejto vrstvy je registrovo orientovaná architektúra, ktorú vykonáva virtuálny stroj Dalvik. Vytvorenie Dalvik-u bolo nutné, kvôli licenčným právam virtuálneho stroja Javy a takisto pre optimalizáciu na mobilné zariadenia, vďaka menšej energetickej

náročnosti vzhľadom na výkon. V tejto vrstve sa tiež nachádzajú základné knižnice programovacieho jazyka Java. Každá spustená Android aplikácia má vlastný proces s inštanciou Dalvikovho virtuálneho stroja. Preklad aplikácie pre Android prebieha skompilovaním Java kódu do Java byte kódu, ten sa ďalej prekompiluje v Dalvikovom kompilátore a Dalvik byte kód je spustený na virtuálnom stroji.

4. vrstva – Application Framework

Zahrňuje programy, ktoré spravujú základné funkcie mobilných zariadení. Do tejto vrstvy majú vývojári aplikácii pre Android úplný prístup, čo im významne napomáha pri možnostiach spracovania a podpore doplnkov pri tvorení aplikácie. Poskytuje základné nástroje, ktoré môže vývojár použiť na tvorbu omnoho komplexnejších.

5. vrstva – Základné aplikácie

Tvoria ju aplikácie pre bežných užívateľov. Jedná sa o predinštalované aplikácie, ale aj dodatočne stiahnuté napr. z Google Play. Sú to: adresár, e-mailový klient, kalendár, mapy, internetový prehliadač a iné aplikácie od tretích strán.

1.3 Verzie

Tvorcovia sa snažia neustále zlepšovať prostredie Androidu a preto vyvíjajú jeho nové verzie. Tu je prehľadná tabuľka momentálne používaných verzii aj s ich menom, ktoré je vždy odvodené od názvu dezertu. Číslo API udáva poradie, v akom boli chronologicky vydávané a je dôležité najmä pre vývojárov. Podiel používanosti sa vzťahuje k dátumu 1. 5. 2014 (Tabuľka 1) [3].

Tabuľka 1: Používané verzie Androidu [3]

Verzia	Názov	API	Podiel
2.2	Froyo	8	1,0%
2.3.3-2.3.7	Gingerbread	10	16,2%
3.2	Honeycomb	13	0,1%
4.0.3- 4.0.4	Ice Cream Sandwich	15	13,4%
4.1.x	Jelly Bean	16	33,5%
4.2.x		17	18,8%
4.3		18	8,5%
4.4	KitKat	19	8.5%

1.4 Aplikácie

Aplikácie sú základným stavebným kameňom tohto systému pre užívateľov. Poskytujú široké spektrum možností využitia telefónu od organizéra a hier až po interné firemné softvéry. Preto sa kladie veľký dôraz na možnosť distribúcie aplikácii. Príkladom sú internetové obchody

Google Play, Amazon Appstore a mnohé iné. Možnosť priamej inštalácie sa realizuje pomocou .apk súboru, ktorý je po stiahnutí možné v zariadení nainštalovať. Nie všetky aplikácie sú dostupné pre užívateľov zdarma. Ceny platených sa pohybujú rádovo v jednotkách dolárov.

1.5 Bezpečnosť a súkromie

Momentálne najväčším problémom tejto platformy sú práve súkromie a bezpečnosť. Aplikácia si pri stiahnutí napr. cez Google Play vyžiada potrebné povolenia, ktoré v mobilnom zariadení musí dostať, aby mohla byť nainštalovaná, v opačnom prípade sa inštalácia nevykoná. Problémom je, že aplikácie si môžu žiadať povolenia pre čítanie a odosielanie SMS správ, určenie polohy pomocou GPS, prístup na Internet alebo iné a pri tom ich k svojmu chodu vôbec nepotrebujú.

Pre zariadenia od verzie 4.2.x je v skúšobnej verzii aplikácia Permission Manager [4] alebo jej podobné, vďaka ktoré umožňujú odoberať povolenia od aplikácii, ale nezaručuje ich funkčnosť. Takýmto aplikáciám sa Android bráni a pravidelne ich z dostupných úložísk odoberá aj keď drvivá väčšina škodlivých aplikácií sa nesnaží si vytvárať povolenia alebo meniť niečo vo svojom sandboxe, ale je pre ne úplne dostačujúce využívať povolenia, ktoré užívateľ pri inštalácií prijme.

1.6 Budúcnosť a vývoj

Pri rýchlosti vývoja tejto platformy môžeme očakávať, že inovácie budú stále zjednodušovať náš každodenný život, od otvárania dverí pomocou mobilného zariadenia, cez platby, až po plánovanie nášho denného programu. Najväčšou úlohou operačného systému Android v nasledujúcom období je presvedčiť, že je dôveryhodný a dokonale bezpečný pre všetky druhy komunikácie a platieb, že umožňuje jednoznačnú autentickosť, zaručí pritom anonymitu užívateľa napr. pomocou biometrických údajov a pritom dokáže stále zostať užívateľsky prívetivý, čo je pri 1,5 milióne nových aktivovaných zariadení denne veľmi dôležité [5,6].

2 Eclipse – Integrované vývojové prostredie (IDE)

Eclipse je platforma s otvoreným kódom, ktorá sa zameriava na vývoj, testovanie a optimalizáciu výkonu aplikácií pre rôzne programovacie jazyky. Štandardne podporovaný programovací jazyk je Java, v ktorom je tento program vytvorený. S doplnkovými modulmi je možné programovať vo veľkom množstve programovacích jazykov (C/C++, Haskell, JavaScript, PHP, Python a iné) [7].

2.1 Android development tools (ADT) modul

ADT je modul pre Eclipse IDE, ktorý je vytvorený pre silné integrované prostredie v ktorom je možné budovať aplikácie pre Android. ADT rozširuje možnosti prostredia Eclipse, pre rýchle vytvorenie Android projektov, vytvoriť grafickú časť aplikácie, pridať balíky založené na verzii API, odstraňovať chyby pomocou nástrojov Android software development kit (SDK) a exportovať .apk súbor, pre distribúciu aplikácie. Programovanie aplikácii pre Android v tomto prostredí je silne doporučené. Tvorenú aplikáciu je možné okamžite testovať na reálnom zariadení, ktoré je pripojené k počítaču, na ktorom je vyvíjaná. Ďalšou možnosťou je test na emulátore [8].

2.2 Trieda java.math.BigInteger

Je trieda, ktorá sa používa pri výpočtoch, kde nie je dostatočný rozsah tried Integer (-2147483648, 2147483647) ani Long (-9223372036854775808, 9223372036854775807) teda najmä pri operáciach v modulárnej aritmetike. V klasifikácii triedy je označená ako nemenné ľubovoľné presné celé číslo so znamienkom. Tieto čísla môžu byť až také veľké, ako to dovolí pamäť RAM na jednotlivých zariadeniach. Pre klasické operácie, sú však tieto čísla absolútne nepoužiteľné, vzhľadom na výpočtovú náročnosť jednoduchých operácii, za aké považujeme násobenie a umocnenie. Avšak pre operácie v modulárnej aritmetike sú čísla patriace do tejto triedy vhodné [9, 10].

2.3 Software development kit (SDK)

Softvérový vývojový set je veľmi dôležitou súčasťou ADT modulu, ktorý zaisťuje, že všetky vytvorené aplikácie budú funkčné na rôznych verziách OS Android. Je zložený z modulárnych balíčkov, ktoré môžu byť sťahované jednotlivo, pomocou programu SDK Manager a jednoducho implementované do vývojového prostredia [11].

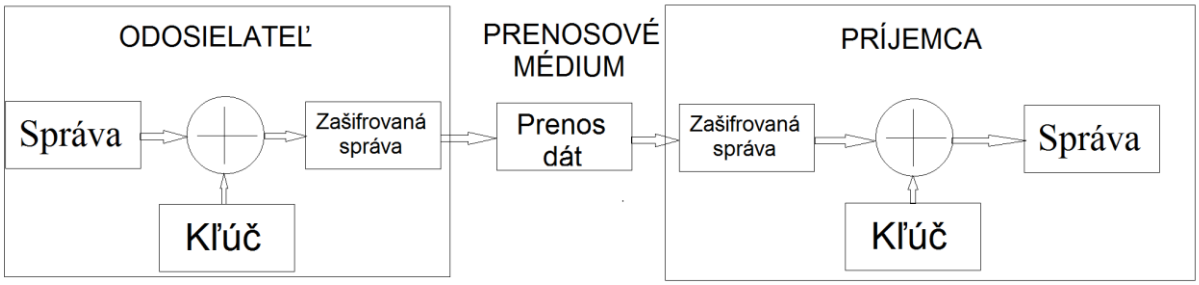
Packages

Name	API	Rev.	Status
Android 4.3 (API 18)			
SDK Platform	18	2	Installed
Samples for SDK	18	1	Installed
ARM EABI v7a System Image	18	2	Installed
Intel x86 Atom System Image	*18*	*1*	*Not installed*
Google APIs	18	3	Installed
Sources for Android SDK	18	1	Installed
Android 4.2.2 (API 17)			
SDK Platform	17	2	Installed
Samples for SDK	17	1	Installed
ARM EABI v7a System Image	17	2	Installed
Intel x86 Atom System Image	17	1	Installed
MIPS System Image	17	1	Installed
Google APIs	17	3	Installed
Sources for Android SDK	17	1	Installed

Obr. č. 1: Android SDK Manager

3 Kryptológia

Slovo kryptológia je zložené z dvoch gréckych slov kryptos = ukrytý, tajný + logos = slovo, náuka, veda. Táto veda sa delí na kryptografiu, ktorá sa zaoberá vytváraním konštrukcii, ktoré z pôvodnej správy vytvoria nezrozumiteľný(zašifrovaný) text, ktorý bude schopný dešifrovať len skutočný príjemca. Naopak kryptoanalýza sa zaoberá nadobudnutím informácií zo zašifrovaného textu bez znalosti kľúča. Kľúč je tajná informácia, ktorej znalosť má iba príjemca a odosielateľ. Nutnosť šifrovania je v dejinách ľudstva neodmysliteľnou súčasťou a preto sa metódy, ktoré sa v kryptológii používali, postupne zdokonaľovali. V dnešnej dobe je nutné šifrovanie veľmi veľkého množstva dát, ktorými sú: komunikácia na internete, bankové transakcie, autentizácia užívateľov a mnoho iných. Obe časti kryptológie preto majú veľký prísľub do budúcnosti, keďže existuje stále viac informácii, ktoré bude nutné šifrovať a neustále sa budú hľadať metódy ako sa k týmto informáciám dostať a zlomiť šifrovanie, ktorým sú zabezpečené [12, 13] .



Obr. č. 2: Základná schéma šifrovania

3.1 Protokol HM 12

Je systém pre atribútovú autentizáciu, vyvinutý na Fakulte elektrotechniky a komunikačných technológií na Vysokom učení technickom v Brne, použiteľný pre poskytnutie väčšieho súkromia pri overení užívateľských atribútov. Vďaka nej je možné preukázať vlastníctvo niektorého atribútu. Týmito atribútmi môžu byť informácie o veku, národnosti, platnosti vodičského oprávnenia. Pri tomto overení však nie je nikde zverejnená identita užívateľa. Overenie je preto anonymné a ďalšími funkciami prispieva k ochrane užívateľského súkromia, akým je napríklad zneužitie informácie zo strany overovateľa [14] .

Pre zachovanie súkromia by mali schémy pre overenie užívateľských atribútov obsahovať nasledujúce funkcie:

- anonymita - uživateľova identita zostáva ukrytá počas overenia vlastníctva atribútu,
- nevystopovateľnosť - neschopnosť vystopovania vydaného atribútu a jeho vlastníka,
- nespojiteľnosť - overenia jedného uživateľa sú navzájom nespojiteľné,

- selektívne zverejnenie atribútov - uživateľ si môže zvoliť, ktoré atribúty poskytne overovateľovi, len potrebné atribúty sú zverejnené,
- neprenositeľnosť - zabránenie prepožičaniu poverení,
- revokácia - neplatné, stratené alebo ukradnuté poverenia sú odvolateľné,
- identifikácia škodlivých uživateľov - aj keď je vlastníctvo atribútu anonymný proces, môže byť identita škodlivých uživateľov a útočníkov odhalená.

Najmä posledné dve funkcie sú ťažko dosiahnuteľné. Kryptografická schéma HM12 podporuje všetky prezentované funkcie.

Táto schéma je špecifická najmä revokáciou poverení:

- Okamžitá revokácia - nie je nutné čakať, kým vyprší platnosť poverenia a je možné ich odvolať okamžite
- Vydavateľom a overovateľom riadená revokácia - odhaliteľnosť je dostupná pre vydavateľa aj overovateľa atribútov. Ktokoľvek z nich môže začať proces revokácie.
- Overovateľova lokálna revokácia- platní uživatelia nemusia aktualizovať svoje poverenia, ani sťahovať žiadne hodnoty po tom, ako sú niektorí uživatelia revokovaní.
- Výpočtová efektívnosť revokácie - výpočtová zložitosť uživateľovej časti overovacieho protokolu nezávisí na počte revokovaných uživateľov
- Overenie bez pripojenia k sieti - overenie prebieha len medzi uživateľom a overovateľom, nie je žiadna potreba kontaktovať ostatné strany

Revokácia poverení je veľmi dôležitou súčasťou, ale v niektorých prípadoch je nedostatočné len odstrániť užívateľa zo systému. V prípadoch, kde bola spôsobená škoda, musí mať poskytovateľ služby možnosť získať identitu útočníkov. Pre lepšiu ochranu súkromia je možné odvolanie niektorých funkcii.

- Odvolanie nespojiteľnosti - overovateľ má možnosť preveriť správanie uživateľa v minulosti odvolaním nespojiteľnosti, všetky minulé relácie vybraného uživateľa môžu byť preverené bez odhalenia jeho identity
- Odvolanie anonymity - v kritických prípadoch je možné, revokovať anonymitu uživateľa pre prevzatie zodpovednosti za jeho činy

Tieto revokácie musia byť chránené proti zneužitiu. Preto sa na nich musí podieľať vydavateľ atribútov, overovateľ a tretia autorita. Pre zvýšenie bezpečnosti a užívateľskej dôvery môže byť tretia autorita rozdelená použitím skupinového výpočtu, pri ktorom sa výpočet prerozdelí pre minimalizovanie možnosti zneužitia. Užívateľ má tiež právo vybrať si vlastného vydavateľa atribútov medzi komerčnými subjektmi, takže nemusí veriť jednej autorite, ale vybrať si entitu, ktorej najviac dôveruje.

V tejto schéme sú spolu štyri entity. Niektoré z nich majú v držaní tajné kľúče.

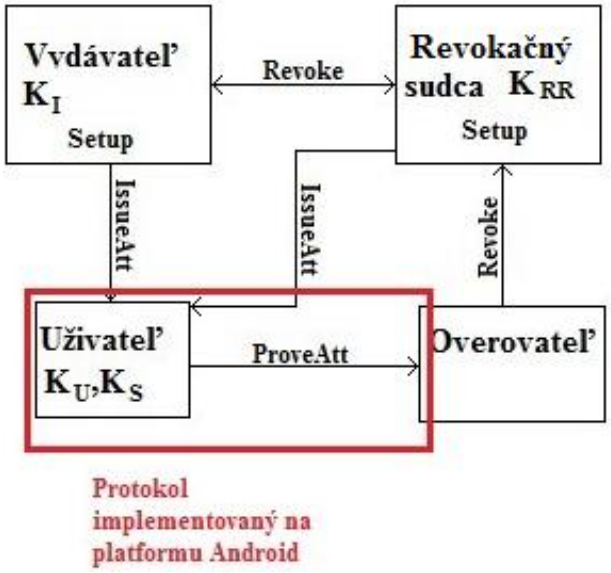
- Vydávateľ (Issuer - I) - entita, ktorá vydáva osobné atribúty uživateľom, vydavatelia tiež spolupracujú počas revokácie anonymity. Vydavateľ, revokačný sudca a overovateľ musia spolupracovať pri odhalení identity škodlivého uživateľa. Vydávateľ je držiteľom kľúča K_1.
- Revokačný sudca (Revocation Referee - RR) - je jediná entita generujúca systémové parametre, spolupracuje s vydavateľom pri vydaní atribútov, s vydavateľom a overovateľom pri odhalení anonymity. Pracuje ako garant súkromia pretože rozhoduje o type odhalenia na základe evidencie poskytnutej overovateľom. Samostatne však nemôže odhaliť žiadne súkromné informácie. Je držiteľom kľúča K_{RR}.
- Uživateľ (User - U) - entita, ktorá je držiteľom zariadenia s vydanými atribútmi. Uživateľ môže anonynmne preukazovať vlastnenie atribútu použitím zariadenia. Je vlastníkom tajného kľúča K_U unikátneho pre každého uživateľa potrebného pre vytvorenie atribútového dôkazu. Navyše kľúč tajnej relácie K_S je generovaný pre každú reláciu overenia. K_S znáhodňuje relácie aby sa stali kompletne nespojiteľné.
- Overovateľ (Verifier - V) - entita, ktorá overuje vlastníctvo užívateľovho atribútu. Použitím prepisu overovacej relácie a dôkazom o porušení pravidiel môže overovateľ požiadať revokačného sudcu o odvolanie. Ak sa sudca rozhodne, že revokácia je legitímna, môže byť uživateľov kľúč K_U anonymne odhalený alebo v serióznejších prípadoch môže byť odhalená identita škodlivých uživateľov. Overovatelia potrebujú iba predzdieľané systémové parametre. Počas overenia užívateľa nekomunikujú s ostatnými stranami (proces funguje bez pripojenia k sieti).

Schéma pozostáva zo štyroch protokolov:

- $(params, K_{RR}, K_I) \leftarrow \text{Setup}(k, l, m)$: je algoritmus medzi sudcom a vydávateľom. Vstupmi Setup sú bezpečnostné parametre (k, l, m) a výstupmi systémové parametre *params*. Sudcov súkromný výstup protokolu je K_{RR} kľúč a vydavateľov súkromný výstup je K_I kľúč.
- $K_U \leftarrow \text{IssueAtt}(params, K_I, K_{RR})$: výstupom protokolu je uživateľský master kľúč K_U. Master kľúč je nutný pre vytvorenie uživateľovho dôkazu o vlastníctve atribútu v ProveAtt protokole. Použitím pokročilých kryptografických techník je K_U generované tak, že sa ho naučí len uživateľské zariadenie aj keď obaja, sudca aj vydávateľ , musia spolupracovať na vytvorení K_U.
- $proof \leftarrow \text{ProveAtt}(params, K_U)$: použitím verejných systémových parametrov a K_U generovaného protokolom IssueAtt je možné vytvoriť atribút *proof* použitím

ProveAtt. Pre každý *proof* je vygenerovaný jednoznačný kľúc relácie K_S. Dôkaz je anonymizovaný a znáhodnený pomocou K_S. Protokol pracuje medzi overovateľom a uživateľským zariadením. Pomocou ProveAtt uživateľ dokazuje vlastníctvo atribútov.

- *rev*←Revoke(*params, proof,* K_{RR}, K_I): v špeciálnych prípadoch (napr. strata, odcudzenie, poškodenie uživateľského zariadenia) , môžu byť vydané atribúty spätne odhalené, alebo môžu byť škodliví uživatelia deanonymizovaní. V tom prípade prepis funkcie *proof* je odoslaný od overovateľa k sudcovi s dostatočnými dôkazmi na odvolanie. Sudca vyhodnotí dôkazy a otvorí prepis *proof* použitím jeho K_{RR}. Podľa typu odhalenia zvoleného sudcom, môže byť atribút zaradený na čiernu listinu, publikovaním anonymnej revokačnej informácie *rev* na verejnú čiernu listinu, alebo poskytne vydavateľovi informácie potrebné k identifikáci uživateľa. Vydavateľ je vtedy schopný identifikovať a postihnúť neoprávneného uživateľa.



Obr. č. 3: Architektúra schémy HM12 [14]

Z hľadiska kryptografie boli v tomto protokole použité:

- záväzky diskrétneho logaritmu - využíva predpoklad, že pre vypočítanie diskrétneho logaritmu veľkého čísla nie je známy žiadny efektívny algoritmus [15].
- Σ-protokoly - môžu slúžiť pre dokázanie vedomosti o tajomstve a pre dokázanie správnej konštrukcie bez úniku ďalších informácií [16].
- Okamoto-Uchiyama jednocestná funkcia - založený na náročnosti faktorizácie veľkých čísel. Doporučenia veľkosti sú rovnaké ako pre RSA kryptosystém [17].
- Hashovacia funkcia - slúži k vyjadreniu rôzne dlhých reťazcov pomocou reťazca konštantnej dĺžky. Jej základnými funkciami sú jednosmernosť(pre vstup je

jednoduché vypočítať výstup, spätne nemožné), jednoznačnosť(každému vstupu prislúcha práve jeden výstup) a rýchlosť(výpočtová nenáročnosť). V mojej implementácií bola použitá hashovacia funkcia SHA-1.

3.2 Detail implementovaného protokolu ProveAtt

USER	PUBLIC PARAMS	VERIFIER
w_1, w_2, w_{rr}	A_{seed}, g_1, g_2, g_3	
$K_s \in_R \{0,1\}^{79}$		
$A = A^{K_s} \bmod n$		
$C_1 = {g_3}^{K_s w_{rr}} \bmod n$		
$C_2 = {g_3}^{K_s} \bmod n$		
$r_1 \in_R \{0,1\}^{480}$		
$r_2 \in_R \{0,1\}^{400}$		
$r_3 \in_R \{0,1\}^{600}$		
$r_s \in_R \{0,1\}^{320}$		
$\overline{A_{seed}} = {g_1}^{r_1} {g_2}^{r_2} {g_3}^{r_3} \bmod n$		
$\overline{A} = A_{seed}^{r_s} \bmod n$		
$\overline{C_1} = g_3^{r_3} \bmod n$		
$\overline{C_2} = g_3^{r_3} \bmod n$		
$e = \mathrm{SHA}-1\,(A, \overline{A}, A_{seed}, \overline{A_{seed}}, C_1, C_2, \overline{C_1}, \overline{C_2})$		
$z_1 = r_1 - eK_s w_1$		
$z_2 = r_2 - eK_s w_2$		
$z_3 = r_3 - eK_s w_{rr}$		
$z_s = r_s - eK_s$		
$\xrightarrow{\quad A, C_1, C_2, e, z_1, z_2, z_3, z_s \quad}$		
		$\overline{A_{seed}} = A^e g_1^{z_1} g_2^{z_2} g_3^{z_3} \bmod n$
		$\overline{A} = A^e A_{seed}^{z_s} \bmod n$
		$\overline{C_1} = C_1^e g_3^{z_3} \bmod n$
		$\overline{C_1} = C_2^e g_3^{z_3} \bmod n$
		$e \stackrel{?}{=} \mathrm{SHA}-1\,(A, \overline{A}, A_{seed}, \overline{A_{seed}}, C_1, C_2, \overline{C_1}, \overline{C_2})$

4 QR kódy

QR kódy boli navrhnuté a vytvorené spoločnosťou Denso Wave so sídlom v Japonsku. QR je skráteným výrazom pre „Quick Response“, čo v preklade znamená rýchla odpoveď. Sú to dvojdimenzionálne štvorcové čiarové kódy (dátové matice), ktoré boli vytvorené pre vysokorýchlostné dekódovanie. V dnešnej dobe sú celosvetovo rozšírené, najmä kvôli propagácií rôznych produktov, pretože môžu na reklamných letákoch priniesť množstvo informácií, ktoré si potenciálny zákazník zosníma a podľa povahy obsahu QR kódu, ho môžu presmerovať na internetovú stránku, ktorá produkt ponúka, alebo priamo ukázať napr. zloženie produktu. Rozmachu QR kódov jednoznačne napomohol rozvoj inteligentných telefónov, ktoré sú vďaka rôznym snímacím aplikáciám schopné tieto kódy dekódovať a odkryť ich obsah. V súčasnosti sa používa QR kód Model 2, v praxi sa vždy pod pojmom QR kód skrýva tento model. V programovacom jazyku Java je pre Android pripravená knižnica com.google.zxing, ktorú je len nutné prispôsobiť si pre potreby aplikácie[18].

Obr. č. 4: QR kód - Hello World

4.1 Kapacita dát

Pri ohľade na snímateľnosť QR kódov a zabezpečenie proti chybovosti je nemožné aby niesli veľmi veľké množstvo informácií, preto sú pre jednotlivé typy udané maximálne množstvá dát (Tabuľka 2).

Tabuľka 2: Maximálna kapacita rôznych typov dát pre QR kódy[18]

Typ vstupných znakov	Maximálna veľkosť
Čísla	7089 znakov
Písmená a čísla	4296 znakov
Binárne (8 bitov)	2953 Bajtov
Japonské znaky Kanji	1817 znakov

4.2 Korekcia chýb

V tematike QR kódov znamená úroveň korekcie chýb hodnotu, akú je ešte schopný snímač QR kódov opraviť a správne dekódovať jeho obsah. Túto vlastnosť je možné si pri vytváraní QR kódu zadať, je pri nej nutné brať ohľad na to, či sa bude nachádzať na mieste, kde môže byť poškodený, z akej vzdialenosti bude snímaný, či prehnane veľká miera úrovne korekcie chýb nenavýši neúmerne jeho veľkosť a nebude tak pre snímače nedekódovateľná, keďže knižnice, ktoré sa na to používajú majú vo svojich dokumentáciách popísané maximálne rozmery, ktoré ešte dokážu dekódovať. Na túto ochranu sa používajú Reed-Solomonove kódy, ktoré pridajú do dát redundanciu, vďaka ktorej je možné ich spätne opraviť. V praxi sa využívajú 4 úrovne korekcie chýb, ktoré sú na každom vygenerovanom QR kóde jednoznačne určené dvojicou bitov(bodov). Pre ilustráciu poslúži QR kód URL adresy: http://www.feec.vutbr.cz/fakulta/home.php.cz (Obr. č. 5) [19]:

- úroveň L
  - úprava poškodenia do 7%,
  - v súčasnosti je doporučeným,
  - určujúce 2 bity: čierny (horný), čierny(dolný).
- úroveň M
  - úprava poškodenia do 15%,
  - často používaný na reklamných plochách,
  - určujúce 2 bity: biely (horný), čierny(dolný).
- úroveň H
  - úprava poškodenia 25%,
  - používaný na ľahko poškoditeľných miestach,
  - určujúce 2 bity: čierny (horný), biely (dolný).
- úroveň Q
  - úprava poškodenia 30%,
  - pri väčšom množstve dát priveľká redundancia,
  - určujúce 2 bity: biely (horný), biely (dolný).

Obr. č. 5: určenie úrovne korekcie chýb QR kódov

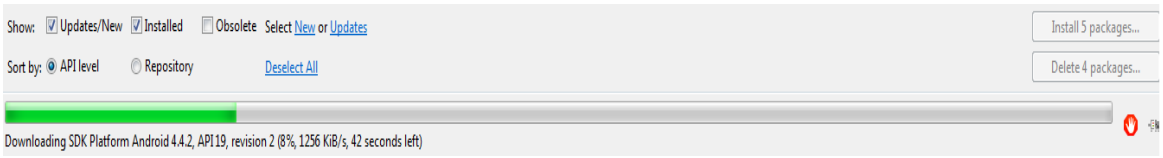
5 Praktická časť - tvorba aplikácie pre Android

Snahou zakladateľov operačného systému Android je, aby preňho mohlo jednoducho vyvíjať aplikácie veľké množstvo programátorov. Aj preto sú manuály na oficiálnych stránkach veľmi dobre popísané. Pre jednoduchosť bude v praktickej časti vysvetlené, ktoré komponenty slúžia k správnemu behu programovania a ako s nimi zaobchádzať.

5.1 Inštalácia a zoznámenie sa s vývojovým prostredím

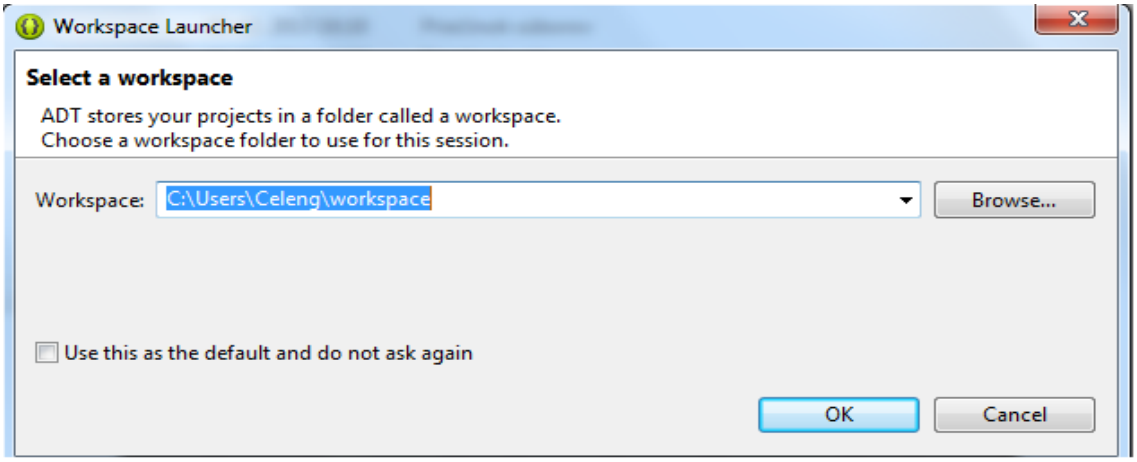
1. Oficiálne vývojové prostredie je voľne stiahnuteľné z oficiálnej stránky http://developer.android.com/sdk/index.html#download , kde je možné vybrať si program pre 32 alebo 64-bitovú verziu softvéru.

2. Po stiahnutí nie je nutná inštalácia, len rozbalenie,avšak nie je možné ani okamžite začať s programovaním. Potrebujeme ešte stiahnuť balíčky z programu Android SDK Manager, ktorý je súčasťou stiahnutého a rozbaleného priečinka. Nevýhodou tejto časti je pomerne veľká nevyrovnanosť serverov, kde sa môže pri niektorých blížiť rýchlosť sťahovania k nule(Obr. č. 6).



Obr. č. 6: SDK Manager sťahovanie balíčkov

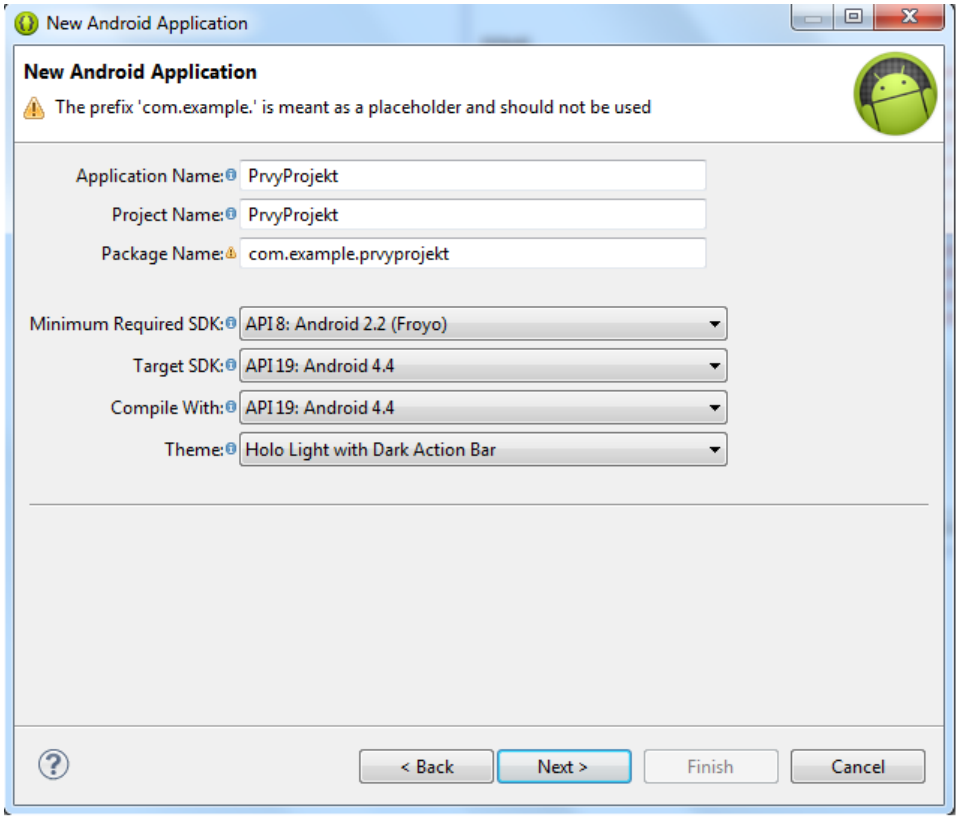
3. V tomto kroku je potrebné zvoliť si adresár, do ktorého sa budú vytvorené aplikácie a ich časti ukladať (Obr. č. 7).



Obr. č. 7: Voľba adresára

4. Nový projekt vytvoríme v menu File →New → Android → Android Application Project.

5. Po tomto zvolení sa otvorí okno, v ktorom zvolíme základné parametre našej budúcej aplikácie: Názov aplikácie, názov projektu, názov balíka, ďalej minimálnu (najnižšia, na ktorej bude aplikácia pravdepodobne fungovať, pri zvolenej API8 je pokrytých viac ako 95% zariadení) a cieľovú verziu zariadenia(všeobecná konvencia hovorí, že to má byť najvyššia dostupná verzia), kompilátor(taktiež sa volí najvyššia verzia) a tému(pre prehľadnosť Holo Light with Dark Action Bar)(Obr. č. 8).

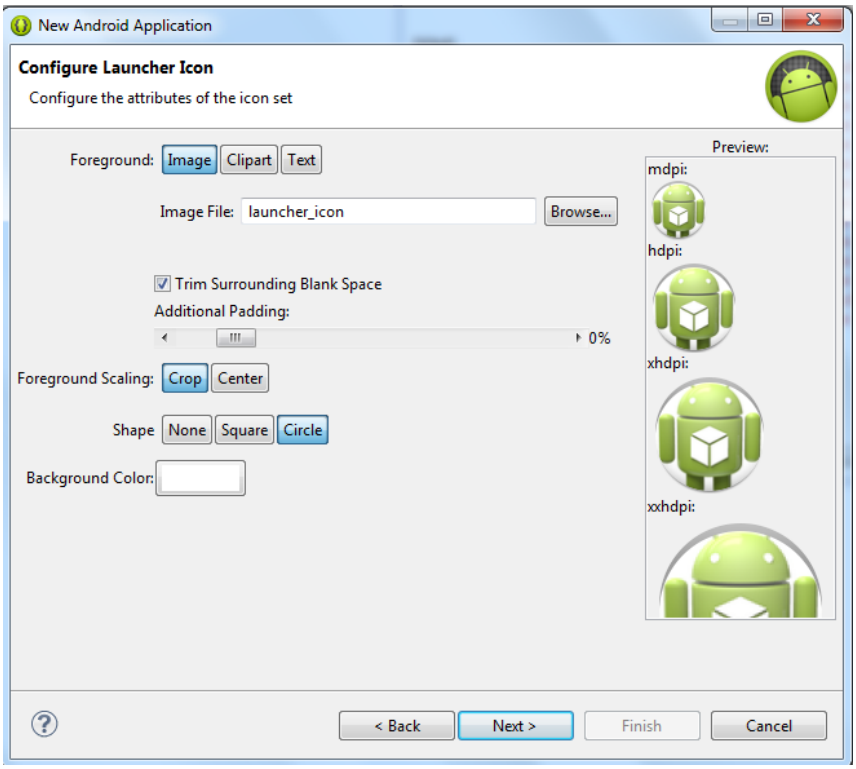


Obr. č. 8: Základné parametre aplikácie

6. Po dvojnásobnom kliknutí na tlačidlo „Next >“ môžeme zvoliť ikonu ktorá bude zobrazená v hlavnej ponuke aplikácií na telefóne(Obr. č. 9).

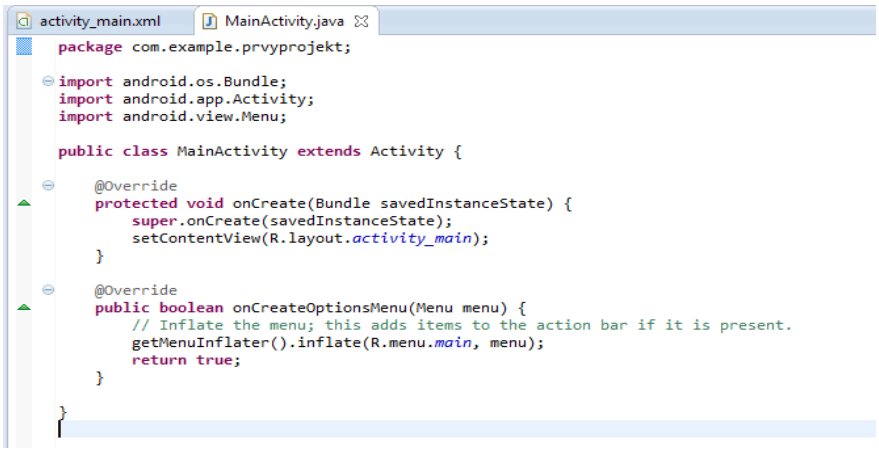
7. V ďalšom okne si môžeme vybrať typ aktivity. Doporučený je momentálne Master/Detail Flow, kvôli kompatibilite s tabletmi, ale ten je možný použiť až od API 11. Preto od verzie API 8 použijeme Blank Activity.

8. V nasledujúcom okne sa volí názov aktivity, názov layoutu a typ navigácie aplikácie.



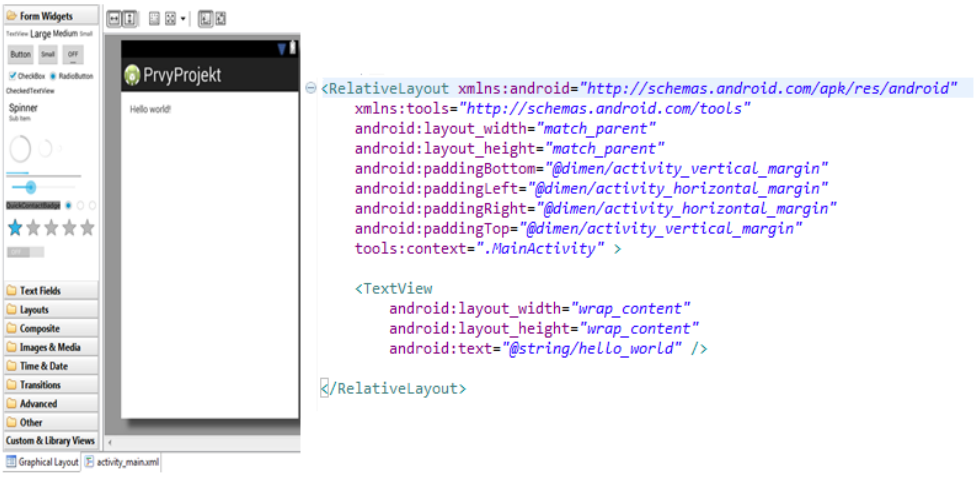
Obr. č. 9: Výber ikony aplikácie

9. Po dokončení tlačidlom Finish sa zjaví súbor MainActivity.java, do ktorého je možné vkladať kód v jazyku Java, ktorý sa bude vykonávať v nami vytvorenej aplikácii(Obr. č. 10).



Obr. č. 10: MainActivity.java

10. Rôzne aktívne prvky ako tlačidlá, textové polia, zaškrtávacie políčka sa pridávajú v layoute. Základný predvytvorený layout má v aplikácii názov activity_main.xml(ak nebol zvolený inak). Prvky je možné pridávať v grafickom rozhraní, alebo priamo vpisovaním kódu do súboru .xml(Obr č. 11).



Obr. č. 11: Xml súbor graficky/kód

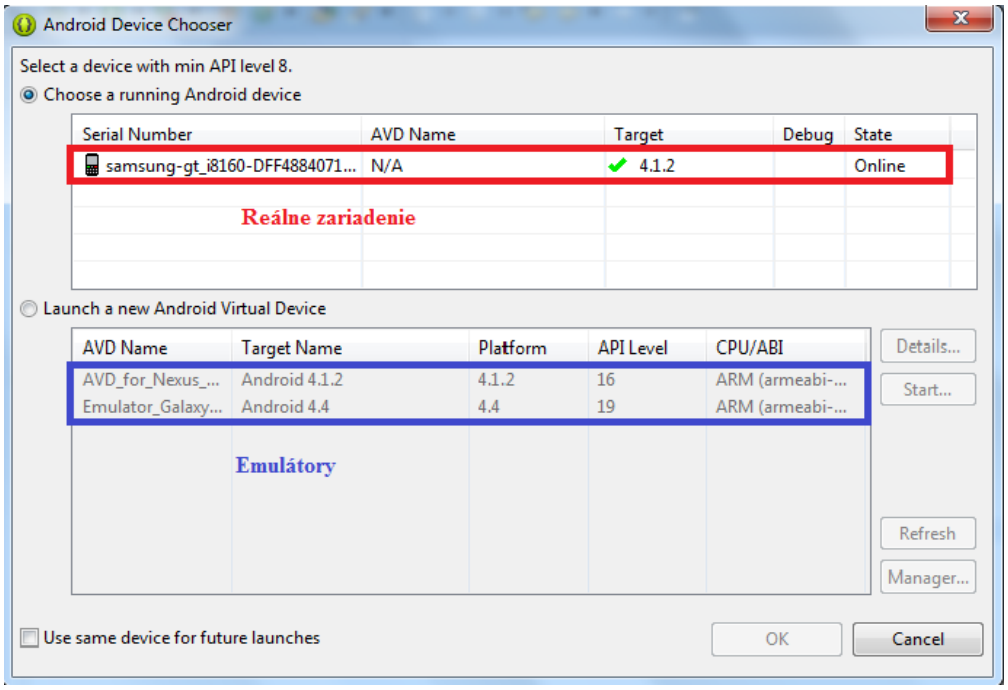
11. Ďalšou dôležitou súčasťou pri vytváraní aplikácie pre Android je Manifest, v ktorom sa nachádzajú dôležité informácie o aplikácii, povolenia, ktoré si musí aplikácia pred nainštalovaním do zariadenia vyžiadať a ak nebudú akceptované, tak sa aplikácia nenainštaluje.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.prvyprojekt"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="19" />  Povolenie pre prístup na Internet

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

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.prvyprojekt.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

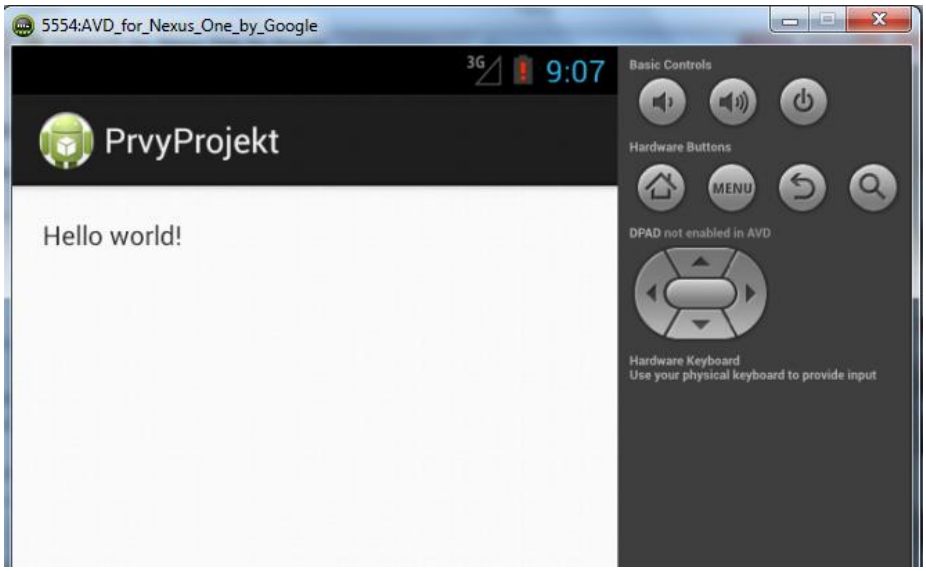
Obr. č. 12: Android Manifest

12. Testovanie vytvorenej aplikácie je možné dvomi spôsobmi a má rôzne výhody:

- Emulátor - môžeme vytvoriť emulátor s rôznymi hardvérovými vlastnosťami, výkonom, rozmerom displeja, verziou Androidu
- Reálne zariadenie - jeho výhodou je najmä rýchlosť a schopnosť častého upravovania a prispôsobovania vytvorenej aplikácie, preto sa aj v praxi preferujú tieto zariadenia



Obr. č. 13: Výber testovacieho zariadenia



Obr. č. 14: Aplikácia zobrazená na emulátore

5.2 Vytvorenie aplikácie pre meranie doby modulárnych operácií

Úlohou tejto práce bolo vytvorenie aplikácie, ktorá zmeria dobu potrebnú pre:

- Generovanie náhodných čísel s dĺžkou 1024, 1536 a 2048 bitov
- Modulárne násobenie čísel s dĺžkou 1024, 1536 a 2048 bitov
- Modulárne mocnenie čísel s dĺžkou 1024, 1536 a 2048 bitov
- ProveAtt protokol s dĺžkou modula 1024 a 2048 bitov

Testovanie bolo primárne vykonávané na telefóne Samsung Galaxy ACE 2 (GT-i8160), s verziou Android4.1.2, ktorého dôležité hardvérové parametre sú:

- Dvojjadrový procesor 800MHz, ARM Cortex-A9
- Grafická jednotka ARM Mali-400MP
- 768 MB pamäte RAM (dostupných 555 MB)

Na ostatných použitých testovacích zariadeniach boli zistené minimálne rozdiely.

Meranie doby

Na meranie času je možné používať preddefinované funkcie v Jave. V tejto práci bola používaná funkcia System.nanoTime(); ktorá sa nevzťahuje k žiadnemu inému časovému údaju a jej rozsah je dostatočne veľký pre merané funkcie. Všetky operácie boli vykonané stokrát a následne bola vypočítaná ich priemerná hodnota a smerodajná odchýlka. Výsledky sa nachádzajú v Tabuľke 3.

Generovanie náhodných čísel

Táto operácia bola vykonávaná veľmi rýchlo, v okolí sto mikrosekúnd a porovnaním týchto meraní bolo zistené, že nezáleží na bitovej veľkosti čísla, ale len na rýchlosti testovania, či vygenerované číslo má skutočne presne preddefinovaný počet bitov.

Modulárne násobenie čísel

Taktiež rýchlo vykonávaná, v stovkách mikrosekúnd, záleží na bitovej veľkosti čísel.

Modulárne mocnenie

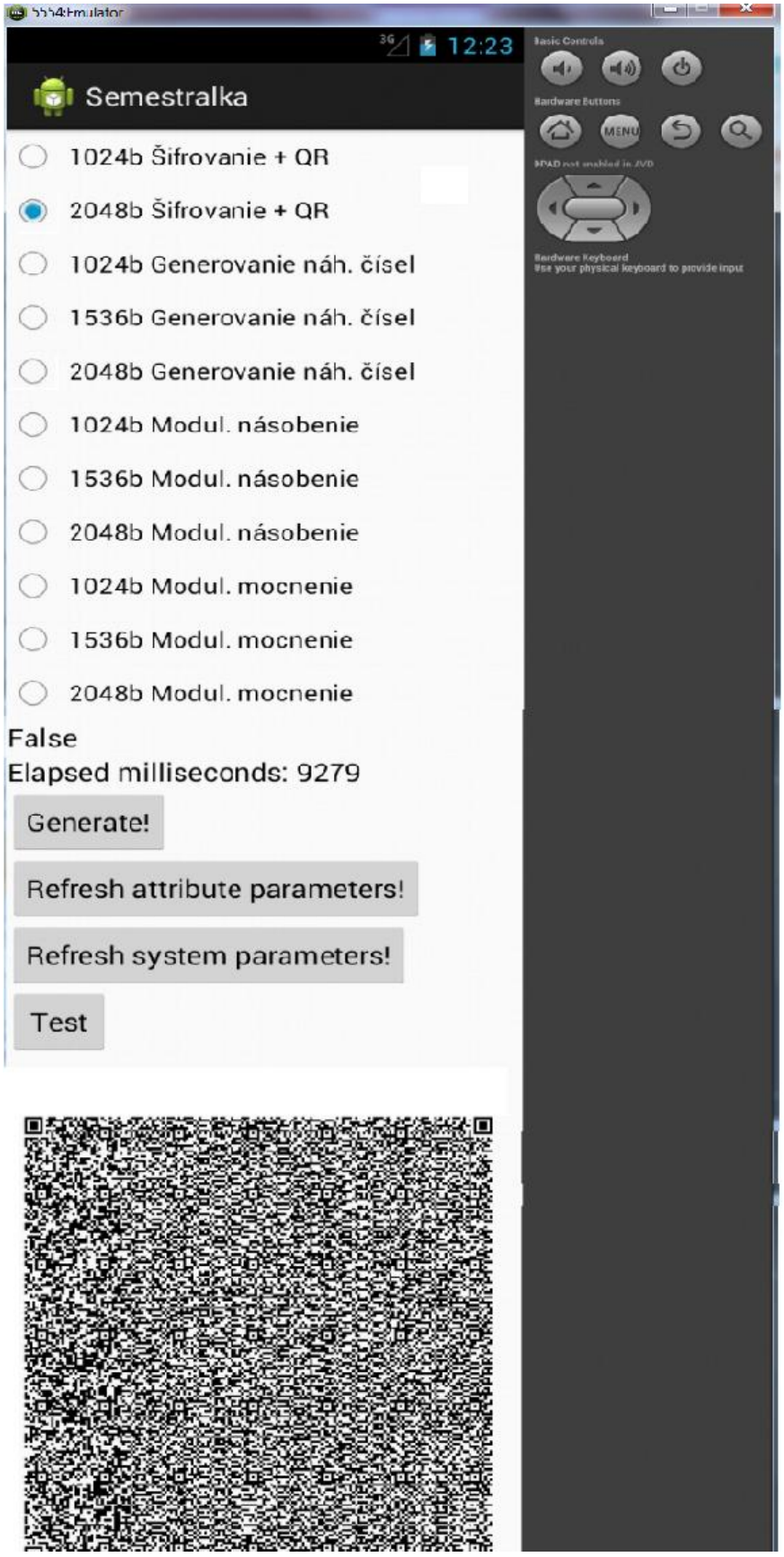
Táto operácia by bola pre triedu BigInteger bez modulárnej aritmetiky úplne nepoužiteľná, avšak s jej použitím sa dosiahne čas rádovo v desiatkach až stovkách milisekúnd pri zadaných bitových dĺžkach. Záleží na bitovej veľkosti čísel.

ProveAtt protokol

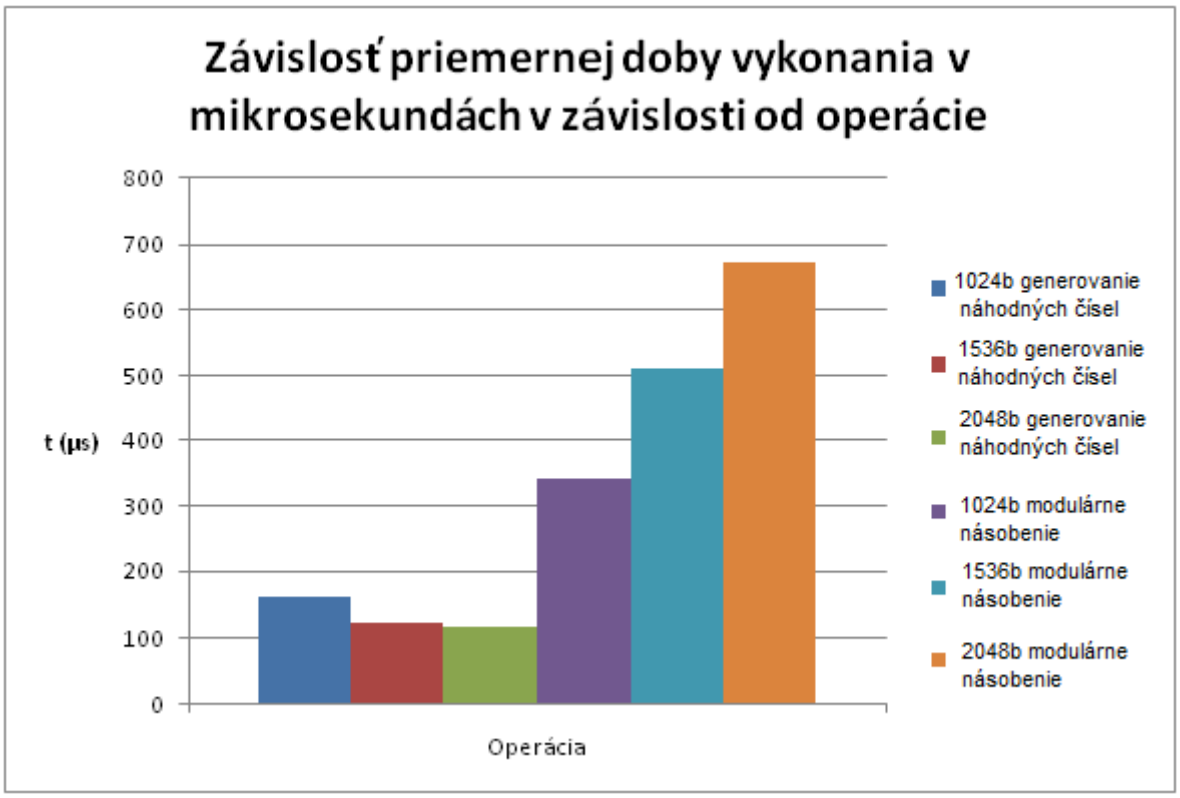
V tomto protokole sa nachádza 9 modulárnych mocnení, 10 modulárnych násobení a štyri rozdiely. Z praxe vieme, že aby bol tento protokol použiteľný, musí byť jeho časová náročnosť maximálne dve sekundy. Súčasťou merania bolo aj generovanie QR kódu, ktorý zabral značnú časť doby vykonania protokolu.

Tabuľka 3: Výsledné meranie modulárnych operácií

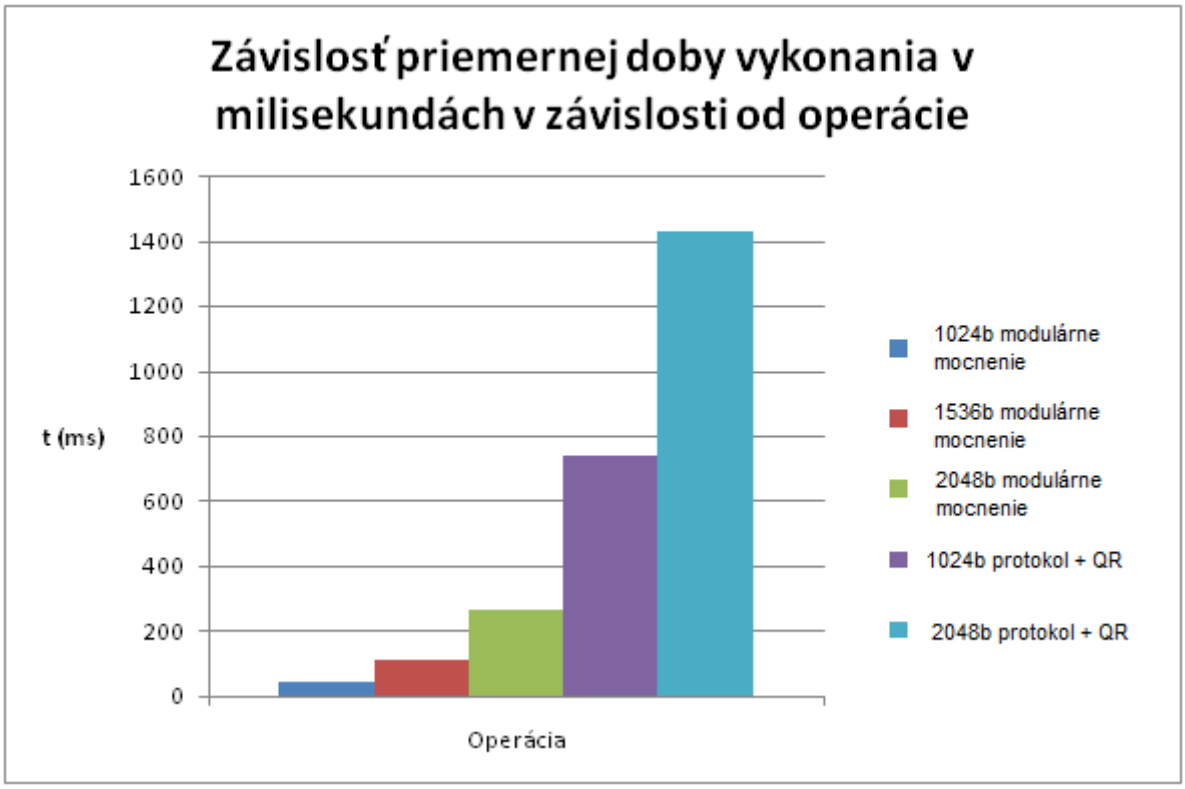
Typ	Čas - t	Priemerná hodnota -$\bar{x}$	Smerodajná odchýlka - σ
1024b generovanie náhodných čísel	[µs]	163,269	140,9353
1536b generovanie náhodných čísel	[µs]	123,9683	131,3512
2048b generovanie náhodných čísel	[µs]	118,4082	61,53361
1024b modulárne násobenie	[µs]	341,7969	235,7174
1536b modulárne násobenie	[µs]	510,8643	1059,26
2048b modulárne násobenie .	[µs]	670,4712	1451,992
1024b modulárne mocnenie	[ms]	43,29	22,15008
1536 modulárne mocnenie	[ms]	107,85	43,94573
2048 modulárne mocnenie	[ms]	266,85	121,8669
1024b implementovaný protokol + QR kód	[ms]	739,37	62,41709
2048b implementovaný protokol + QR kód	[ms]	1430,31	86,45169



Obr. č. 15: Screenshot vytvorenej aplikácie pre meranie časovej náročnosti



Obr. č. 16 Priemerná doba generovania veľkých čísel a modulárneho násobenia



Obr. č. 17 Priemerná doba modulárneho mocnenia a protokolu generujúceho QR kód

5.3 Vytvorenie klientskej aplikácie zadaného protokolu

Najdôležitejšou časťou zadania bolo vytvorenie klientskej aplikácie, ktorá by fungovala v spolupráci s overovacou aplikáciou (vyvíjanou ako súčasť diplomovej práce iného študenta) pre prihlásenie sa na zabezpečenú internetovú stránku. Podstata je v tom, že telefón s OS Android vygeneruje QR kód, v ktorom budú zakódované informácie potrebné k overeniu majiteľa daného atribútu, tie sa pomocou webkamery zosnímajú a následne, po overení na serveri, buď prístup povolí alebo zamietne. Platnosť tohto QR kódu je obmedzená pomocou časovej známky na 120 sekúnd, kvôli zabezpečeniu proti zneužitiu. Požiadavky na klientskú aplikáciu teda sú:

- jednoduchá použiteľnosť a ovládanie
- minimálna časová náročnosť
- dobrá snímateľnosť kódu
- spolupráca s overovacou aplikáciou
- kompatibilita na rôznych zariadeniach používajúcich OS Android
- čo najmenej povolení pri inštalácií aplikácie
- malé riziko zneužitia.

Logickým riešením teda môže byť aplikácia, v ktorej sa po spustení vypočítajú potrebné operácie v modulárnej aritmetike s dĺžkou modula 1024 bitov (kvôli nižšej časovej náročnosti a dostatočnej bezpečnosti), tie sa následne podľa požiadaviek protokolu zlúčia do jedného znakového reťazca z ktorého sa následne vygeneruje požadovaný QR kód, ktorý sa vykreslí v aplikácií a môže byť zosnímaný. Pre túto implementáciu bola zvolená knižnica com.google.zxing, ktorá je dostupná pre zariadenia Android, je komunitou odporúčaná, neustále programátorsky podporovaná, vydáva nové aktualizácie a najmä aplikácie založené na báze tejto knižnici sú najúspešnejšími komerčnými aplikáciami v rámci tejto problematiky.

Po implementácii a naprogramovaní aplikácie ktorá pracuje s touto knižnicou a vykonáva požadované operácie sme získali QR kód, v ktorom boli podľa dohody čísla v desiatkovej sústave a oddeľovacie znaky „_“ medzi jednotlivými číslami. Výsledný QR kód bol verzie 21 (čo znamená, že kód má 22 ochranných štvorčekov). Skontrolované boli všetky atribúty, ktoré mohli túto verziu ovplyvniť: úroveň opravy chýb nastavená na L (nízku), žiadne prebytočné dáta v implementovanom protokole, vybraná znaková sada ISO-8859-1 taktiež nemala žiadny vplyv.

Na telefóne s OS Android nebol problém generovaný QR kód takejto veľkosti (verzie) dekódovať. Problémom však bola spolupráca s overovacou aplikáciou. Tá bola vytvorená na programovacej platforme C# taktiež s použitím knižnice com.google.zxing. Tá ale nedokáže pri snímaní pomocou vstavanej ani externej webkamery dekódovať QR kódy tejto veľkosti, pravdepodobne preto, že táto súčasť nie je zatiaľ v praxi veľmi využívaná a do QR kódov sa namiesto dlhých reťazcov vkladajú URL adresy alebo len stručné informácie o produktoch. Je ale otázkou času, kedy začne táto knižnica dekódovať ľubovoľne veľké QR kódy a vtedy sa stane vytvorená aplikácia najlepšou možnou variantou pre užívateľov. Pre jej rýchlosť, nezávislosť povolení pri inštalácii (nepotrebuje žiadne) a dobrú bezpečnosť pri štandardných zariadeniach.

Dovtedy je nutné nájsť inú možnosť pre klientskú aplikáciu. Riešení tejto situácie je hneď niekoľko:

- Nájdenie inej knižnice
- Vytvorenie dvoch menších QR kódov namiesto jedného veľkého
- Použitie externej API dostupnej na Internete pre vytvorenie QR kódu

Nájdenie inej knižnice

Na Internete je možné nájsť aj iné knižnice riešiace túto problematiku, ale drvivá väčšina z nich vychádza z knižnice, ktorá bola implementovaná mnou a neupravuje jej podstatu (čo reálne ani nemôže), iba pridáva ďalšie funkcie, alebo zjednodušuje implementáciu avšak za cenu zvýšenia potrebnej pamäte, čo je pre aplikáciu nežiaduce. Najväčším prísľubom do budúcnosti je knižnica net.sourceforge.zbar.android ktorá je taktiež s otvoreným kódom, ale momentálne ponúka len dekódovanie QR kódov (s výbornými vlastnosťami) v jej návrhu je tiež QR Encoder, ktorý bude QR kódy generovať. Ďalšie knižnice pracujú tak, že vykresľujú QR kódy ako obrázky a potrebujú rôzne povolenia od užívateľa pre svoj beh a sú taktiež nefunkčné pre takéto množstvo dát. Záverom je, že momentálne možnosť nájdenia inej knižnice nie je pre implementáciu možná.

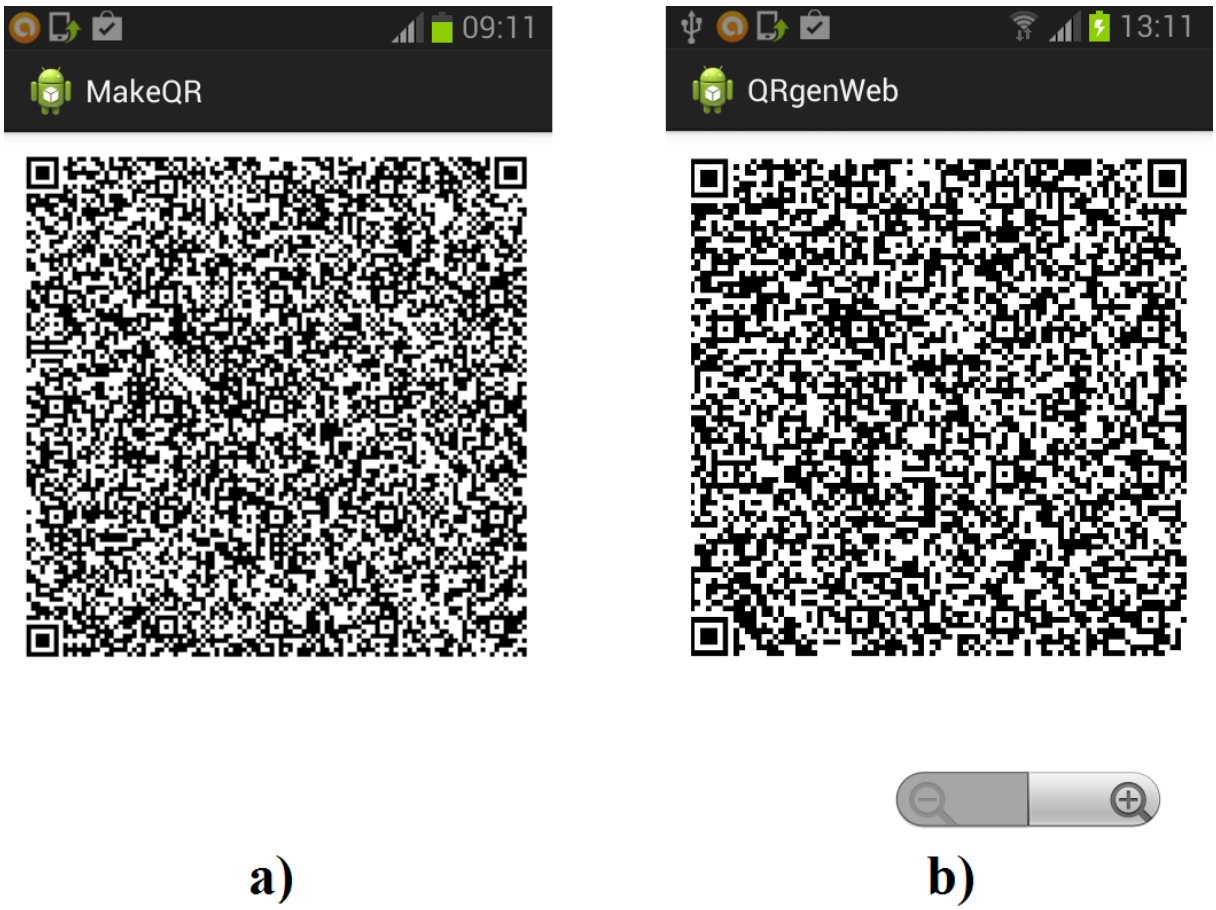
Vytvorenie dvoch menších QR kódov namiesto jedného veľkého

Vygenerovaný QR kód nie je možné overovacou aplikáciou prečítať. Preto je jednou z možností rozdeliť dáta na dve polovice a vygenerovať dva menšie QR kódy, ktoré bude overovacia aplikácia môcť dekódovať a overiť. Táto možnosť je najlogickejšou a programátorsky pre klientskú aplikáciu najjednoduchšie dosiahnuteľnou. Nie je ale prijateľná pre jednoduchosť ovládania klientov. Je to zdvojnásobenie nutnosti skenovania QR kódu, na rôznych zariadeniach by sa nesprávala rovnako, napríklad pri zariadeniach s menším displejom by nemohli byť QR kódy vedľa seba, ale pod sebou, bolo by nutné sa v aplikácii

posúvať alebo v úplne najhoršom prípade by sa zosnímal jeden QR kód, overovacia aplikácia by vydala napr. zvukový signál, užívateľ by stlačil tlačidlo, ktorým by sa vykreslil druhý kód, tam kde bol predtým prvý. S takouto aplikáciou by užívatelia určite neboli spokojní. Ďalšou možnosťou by bola zmena overovacej aplikácie tak, že by čítala dva QR kódy zároveň, čo zatiaľ žiadna knižnica nepodporuje a preto by mohlo dochádzať k chybám pri postupnom čítaní, nehovoriac o náročnosti takejto implementácie. Toto riešenie teda nie je užívateľsky vôbec vhodné a preto je potreba nájsť inú možnosť.

Použitie externej API dostupnej na Internete pre vytvorenie QR kódu

Pre túto možnosť je potrebné nájsť vhodnú API- Application Programming Interface (rozhranie pre programovanie aplikácií), ktorá veľmi rýchlo dokáže po zadaní informácií a potrebných dát, poslať späť tieto dáta v podobe QR kódu. Z pohľadu bezpečnosti bolo nutné nájsť server, ktorý je schopný tieto informácie posielať pomocou zabezpečeného štandardu https. Ďalej bolo nutné zistiť ako kódovaný obrázok zobraziť na telefóne. V princípe existujú dve možnosti. Prvou je stiahnutie si QR kódu do pamäte telefónu a následne ho zobraziť. Na tieto úkony by sme teda dokopy potrebovali povolenie od užívateľa na prístup k Internetu, zapisovanie do pamäte a čítanie z pamäte. Druhou (použitou možnosťou) bolo zobrazenie QR kódu pomocou komponentu WebView, ktorý je určený pre zobrazenie internetových stránok, pričom užívateľ vizuálne ani nemá ako zistiť, že sa nachádza na takejto stránke. Pre istotu použiteľnosti na všetkých zariadeniach bola pridaná možnosť priblíženia a oddialenia QR kódu. Vzhľad tejto aplikácie je rovnaký ako aplikácie s použitím knižnice com.google.zxing, avšak generovaný QR kód je vyhovujúci pre overovaciu aplikáciu a plne s ním spolupracuje (je teda možné overiť pomocou neho užívateľa a prihlásiť ho na želanú stránku.). Formát odosielaný na API server je: *api.qrserver.com/v1/create-qr-code/?size=1000x1000&charset-source=ISO-8859-1&charset-target=ISO-8859-1&data=dáta ktoré majú byť vložené do QR kódu.* V parametri size je rozmer, ktorý vykreslí kód na celú šírku WebView, keďže QR kódy sú štvorcové, musia byť oba parametre rovnaké, charset je označenie pre použitú znakovú sadu, v tomto prípade bola použitá aj pre zdrojové aj výstupné dáta použitá sada ISO-8859-1, do kolónky data sa vkladajú dáta pre vloženie do QR kódu (v tomto prípade výstup z autentizačného protokolu). Pre lepšiu ilustráciu sú obe aplikácie na Obr. č. 18 vedľa seba. Úspora dát medzi generovanými QR kódmi je zjavná, rozdiel medzi verziami 14 a 21 je spolu 4872 bodov. Za zmienku, ale taktiež stojí aj bezpečnostné hľadisko, keďže dáta je nutné odoslať cez internet na cudzí server, čomu by bolo vhodné sa v budúcnosti vyhnúť, pretože aj vygenerovaný QR kód sa neprenáša bod po bode ale ako internetová stránka, to by mohol využiť potenciálny útočník a dostať sa ku QR kódu s autentizačnými dátami omnoho jednoduchšie [20,21].



Obr. č. 18 Screenshot vytvorenej klientskej aplikácie generovanej a) pomocou knižnice, b) pomocou API

6 Záver

Cieľom bakalárskej práce bolo zoznámenie sa s vývojovým prostredím pre operačný systém Android, ktorý je veľmi perspektívnym a dobre podporovaným prostredím pre vývojárov aplikácií. Ďalej bolo nutné pochopiť význam bezpečnosti a trendy, ktoré sa v modernej kryptografii dostávajú do popredia, význam veľkých čísel pri ich implementáciách, prácu s týmito číslami v modulárnej aritmetike a časovú náročnosť operácií v nej.

Hlavným zadaním bolo vytvorenie funkčnej aplikácie, ktorá bude schopná zmerať časovú náročnosť jednotlivých operácií potrebných pre implementáciu protokolu pre atribútové overenie a následne aj daný protokol, doposiaľ implementovaný na čipových kartách, ktorého výstupom je QR kód, nesúci potrebné informácie pre overenie. Na začiatku bolo nutné oboznámiť sa s vývojovým prostredím Androidu, jeho inštaláciou, funkciami, ktoré ponúka pre prácu s veľkými číslami (trieda BigInteger) a taktiež aj s generovaním QR kódov. V teoretickej príprave som sa zameral na operačný systém Android ako taký, pochopenie potreby šifrovania, na implementovaný protokol HM12 z dielne pracovníkov FEKT VUT a rôzne vlastnosti QR kódov.

Zadanie sa podarilo splniť, meracia aplikácia je plne funkčná, priemerné výsledky meraní a smerodajné odchýlky pre potrebné funkcie sú uvedené v tabuľke (Tabuľka 3) a grafoch (Obr. č. 16 a Obr. č. 17) z ktorých vyplýva, že táto implementácia, by mohla byť vzhľadom na časovú náročnosť pokojne využívaná v praxi, keďže moderné zariadenia majú dostatočný výpočtový výkon pre potreby kryptografických aplikácií.

V ďalšej časti bakalárskej práci bola mnou vytvorená klientská aplikácia, ktorá odovzdáva overovacej aplikácii (vypracovaná ako diplomová práca študenta Petra Vybírala) snímateľné dáta prostredníctvom QR kódu a má parametre aplikácie importovateľnej do praxe. Bolo pre mňa veľmi cennou skúsenosťou vyskúšať si prácu v tíme.

Prísľubom od takéhoto pokroku do budúcnosti je možnosť, že namiesto množstva čipových kariet, ktoré musí mať v dnešnej dobe pri sebe bežný človek, mu bude stačiť jeho mobilný telefón, čo by mu malo uľahčiť jeho každodenný život, zjednodušiť platby či overenia pri vstupe do budov a veľa ďalších dobrých funkcií pri zachovaní vysokého stupňa anonymity. Je ale potrebné zamedziť zneužívaniu týchto funkcií, cielenej kriminalite či už softvérovej alebo hardvérovej voči mobilným zariadeniam. Na to je nutné zlepšenie bezpečnosti mobilných zariadení a vyššia zodpovednosť užívateľov.

Zoznam použitých zdrojov

[1] A brief history of Android. DELEON, Walter. Technoblimp [online]. 2013 [cit. 2013-12-22]. Dostupné z: http://technoblimp.com/2013/09/14/a-brief-history-of-android/

[2] How the Google Phone Works: Google Android Architecture. STRICKLAND, Jonathan. Howstuffworks [online]. 2007 [cit. 2013-12-22].Dostupné z: http://electronics.howstuffworks.com/google-phone2.htm

[3]Dashboards. Android developers [online]. 2013 [cit. 2013-12-22]. Dostupné z: http://developer.android.com/about/dashboards/index.html

[4] Permission Manager. Google play [online]. 2013 [cit. 2013-12-22]. Dostupné z: https://play.google.com/store/apps/details?id=com.gmail.permissionmanager&hl=sk

[5] Android in 2020: the future of Google's mobile OS explored. NIELD, David. Techradar [online]. 2013 [cit.2013-12-22]. Dostupné z: http://www.techradar.com/news/software/operating-systems/android-in-2020-the-future-of-google-s-mobile-os-explored-1168141

[6] Google Reports 1.5 Million Android Activations Per Day. ZEMAN, Eric. InformationWeek [online]. 2013 [cit. 2014-05-25]. Dostupné z: http://www.informationweek.com/mobile/mobile-devices/google-reports-15-million-android-activations-per-day/d/d-id/1109568?

[7]About the Eclipse Foundation. Eclipse [online]. 2013 [cit. 2013-12-22]. Dostupné z: http://www.eclipse.org/org/

[8] ADT Plugin. Android developers [online]. 2013 [cit. 2013-12-22]. Dostupné z: http://developer.android.com/tools/sdk/eclipse-adt.html

[9] Java.math.BigInteger Class. Tutorialspoint [online]. 2013 [cit. 2013-12-22]. Dostupné z: http://www.tutorialspoint.com/java/math/java_math_biginteger.htm

[10] Uses of Class java.math.BigInteger. Docs.oracle [online]. 2013 [cit. 2013-12-22]. Dostupné z: http://docs.oracle.com/javase/7/docs/api/java/math/class-use/BigInteger.html

[11]Exploring the SDK. Android developers [online]. 2013 [cit. 2013-12-22]. Dostupné z: http://developer.android.com/sdk/exploring.html

[12] Cryptology PELL, Oliver. Ridex[online]. 2007 [cit. 2013-12-22]. Dostupné z: http://www.ridex.co.uk/cryptology/#_Toc439908851

[13] Princípy bezpečnej komunikácie. DT certification authority [online]. 2005 [cit. 2013-12-22]. Dostupné z: http://www.dtca.sk/support/principles.php

[14] HAJNÝ, J.; MALINA, L. Unlinkable Attribute-Based Credentials with Practical Revocation on Smart- Cards. In Smart Card Research and Advanced Applications. Lecture Notes in Computer Science. LNCS. Berlin: Springer- Verlag, 2013. s. 62-76. ISBN: 978-3-642-37287- 2. ISSN: 0302- 9743.

[15] Camenish, J., Stadler, M.: Proof systems for general statements about discrete logarithms. Tech. rep. (1997)

[16] Cramer, R.: Modular Design of Secure, yet Practical Cryptographic Protocols. Ph.D. thesis, University of Amsterdam (1996)

[17] Okamoto, T., Uchiyama, S.: A new public-key cryptosystem as secure as factor-ing. In: Advances in Cryptology - EUROCRYPT 98, Lecture Notes in ComputerScience, vol. 1403, pp. 308{318. Springer Berlin / Heidelberg (1998)

[18] About QR code. Mobile-Barcodes [online]. 2013 [cit. 2013-12-22]. Dostupné z: http://www.mobile-barcodes.com/about-qr-codes/#standards

[19] QR code error correction. QR stuff [online]. 2011 [cit. 2014-05-25]. Dostupné z: http://www.qrstuff.com/blog/2011/12/14/qr-code-error-correction

[20] Information capacity and versions of the QR code. DENSO WAVE INCORPORATED. QR code [online]. 2014 [cit. 2014-05-25]. Dostupné z: http://www.qrcode.com/en/about/version.html

[21] QR code generator. GoQR.me [online]. 2014 [cit. 2014-05-25]. Dostupné z: http://goqr.me/api/doc/create-qr-code/

[22] CELENG, M. Bezpečnostní aplikace pro Android a Apple iOS. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2014. 29 s. Vedoucí semestrální práce Ing. Jan Hajný, Ph.D..

Zoznam skratiek a symbolov

OS	Operating system
QR	Quick response
.apk	application package file
IDE	Integrated development enviroment
ADT	Android development tools
SDK	Software development kit
API	Application programming interface
RAM	Random-access memory
URL	Uniform resource locator
ARM	Advanced RISC Machines
https	Hypertext transfer protocol secure

Zoznam príloh

Príloha 1: Zdrojový kód semestrálna práca

Príloha 2: Zdrojový kód bakalárska práca

Príloha 3: Aplikácia Semestralka.apk

Príloha 4: Aplikácia BakalarskaPraca.apk