VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ZABEZPEČENÝ PŘENOS DAT POMOCÍ ČÁROVÝCH KÓDŮ

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MARTIN KRATOCHVÍL

BRNO 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ZABEZPEČENÝ PŘENOS DAT POMOCÍ ČÁROVÝCH KÓDŮ

SECURE DATA TRANSMITION USING BAR CODES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. MARTIN KRATOCHVÍL
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÍTĚZSLAV BERAN, Ph.D.
SUPERVISOR

BRNO 2011

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2010/2011

Zadání diplomové práce

Řešitel: **Kratochvíl Martin, Bc.**
Obor: Počítačová grafika a multimédia
Téma: **Zabezpečený přenos dat pomocí čarových kódů**
Secure data transmition using bar codes
Kategorie: Bezpečnost

Pokyny:

1. Seznamte se s problematikou 1D a 2D čárových kódů a jejich variantami a zabezpečeným přenosem dat.
2. Navrhněte koncepty pro přenos dat s využitím čarových kódů s různou úrovní zabezpečení. Přenos nechť je realizován pomocí zobrazovacího a čtecího zařízení.
3. Navržené koncepty implementujte jako knihovnu funkcí. Využijte existující nástroje pro práci s čarovými kódy i pro šifrování.
4. Navrhněte a popište aplikace využívající tyto koncepty v praxi.
5. Najděte slabá místa řešení a proveďte experimenty zkoumající jejich vliv. Diskutujte výsledky experimentů.
6. Vytvořte plakát reprezentující Vaše řešení.

Literatura:

- Dle pokynů a doporučení vedoucího.

Při obhajobě semestrální části diplomového projektu je požadováno:

- Body 1, 2 a částečně 3.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Beran Vítězslav, Ing.**, UPGM FIT VUT
Datum zadání: 20. září 2010
Datum odevzdání: 25. května 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Cílem práce bylo vytvoření systému pro přenos dat vizuální cestou s využitím čárových kódů. Důraz byl kladen hlavně na zabezpečení systému proti zneužití. Byl navržen mechanizmus pro samotný přenos a různé bezpečnostní koncepty. Na základě analýzi byl pro přenos dat zvolen nejvhodnější čárový kód.

Abstract

The goal of this thesis was to create a system for visual data transmition using bar codes. It focuses mainly on the protection of the system against abuse. A mechanism was designed for the data transmition itself and the various security concepts. The most appropriate bar code for data transmition was selected on the basis of the analysis.

Klíčová slova

QR kód, čárové kódy, bezpečnost, kryptografie, Java, Android, AES, RSA, elektronický podpis

Keywords

QR code, barcodes, security, cryptography, Java, Android, AES, RSA, electronic signature

Citace

Martin Kratochvíl: Zabezpečený přenos dat pomocí čárových kódů, diplomová práce, Brno, FIT VUT v Brně, 2011

Zabezpečený přenos dat pomocí čárových kódů

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Vítězslava Berana

.....
Martin Kratochvíl
25. května 2011

Poděkování

Chtěl bych poděkovat vedoucímu práce Ing. Vítězslavu Beranovi za jeho odbornou pomoc při zpracování práce.

Obsah

Kapitola 1

Úvod

Tématem této práce je zabezpečený přenos dat pomocí čárových kódů. Cílem je navrhnout samotný přenos dat vizuální cestou za využití zvoleného čárového kódu (na základě analýzy). Systém lze využít všude tam, kde jsou používána citlivá data pro přístup do cizích systémů. Aplikace vyvinutá v rámci této práce zajistí funkci bezpečné schránky citlivých dat i bezpečný způsob jejich přenosu do jiných systémů. Největší důraz je kladen na bezpečnost.

Kapitola 2 je věnována přehledu různých typů čárových kódů. Jsou uvedeny vlastnosti jednotlivých kódů a na závěr je uvedeno, které vlastnosti kódu jsou vhodné pro použití v navrhovaném systému.

V následující kapitole (kapitola 3) jsou vysvětleny pojmy týkající se bezpečnosti a hlavně jsou zde popsány principy šifrování. Šifrování je rozděleno do hlavních skupin a ke každé jsou kromě popisu principu uvedeny i stále používané algoritmy.

V Kapitole 4 nazvané Návrh systému je rozepsán samotný navrhovaný systém a také je zde navržen vizuální přenos informací. Kromě toho je v této kapitole rozepsán návrh různých prvků zabezpečení tohoto systému.

Další kapitolou je kapitola 5, která popisuje samotnou realizaci systému. Realizací systému jsou myšleny vyvinuté komponenty v rámci této práce, popis jejich funkce a případná možná rozšíření pro použití v praxi. Krom toho je v kapitole i zmínka o použitých nástrojích usnadňujících realizaci.

V kapitole 6 nazvané Aplikace systému popsáno využití systému v různých reálných oblastech. Kapitola popisuje integraci systému, vhodné parametry a vlastnosti systému, které je nutné v konkrétních případech zohlednit.

Kapitola 2

Čárové kódy

V této kapitole bude uveden přehled několika typů čárových kódů a jejich rozdělení dle vlastností důležitých při vytváření popisovaného systému. Na základě tohoto přehledu bude zvolen nejvhodnější čárový kód, který bude nakonec použit při implementaci systému.

2.1 Co jsou čárové kódy?

Čárové kódy jsou data zakódovaná do podoby čitelné optickými přístroji [16]. Původně měly podobu pouze skupiny černých čar umístěných souběžně a informace byla kódována do rozdílných šířek jednotlivých čar, takovéto kódy se nyní označují jako 1D (jednorozměrné) čárové kódy. Nyní mohou mít různé podoby, například čtverce, tečky, šestiúhelníky a jiné geometrické vzory – ty jsou označovány jako 2D (dvoudimenzionální) maticové kódy. Ačkoli 2D čárové kódy používají jiné symboly než čáry, jsou také obecně označovány jako čárové kódy. Čárové kódy mohou být přečteny pomocí optických skenerů nazývaných čtečky čárových kódů, nebo dekódovány z obrázku pomocí speciálního softwaru. Poprvé se začali používat k označování železničních vagónů, ale nebyli komerčně úspěšné až do doby kdy se začali používat v supermarketech jako automatický kontrolní systém, úkol ve kterém se stali téměř univerzální. Jejich používání se rozšířilo do mnoho dalších oblastí, jejich úkol se obecně označuje jako Auto ID Data Capture (AIDC). Také jiné systémy se pokoušeli prorazit na trh AIDC, ale jednoduchost, univerzálnost a nízká cena čárových kódů to umožnila jen z malé části.

2.2 Popis základních typů - 1D čárové kódy

Nejprve bude uveden přehled některých konkrétních jednorozměrných čárových kódů. Pro všechny 1D čárové kódy platí jedna společná základní myšlenka a to, že jsou sestrojeny ze skupiny černých čar a mezer mezi nimi. Samotná informace se pak kóduje do pořadí střídaní čar a mezer a do šířky jednotlivých čar a mezer. Jednotlivé kódy se potom liší v algoritmu samotného kódování, v množině dat, kterou umožňují zakódovat a v množství dat, které umožňují pojmout.

UPC

UPC (Universal Product Code – univerzální kód výrobku) je 1D čárový kód, jehož použití je široce rozšířeno hlavně v Kanadě a Spojených státech amerických [4]. Používají se především

pro sledování zboží ve skladech. Na obrázku 2.1 je vidět příklad UPC čárového kódu.

Obrázek 2.1: Příklad kódu UPC

UPC čárový kód kóduje 12 číslic ve formátu SLLLLLLMRRRRRRE, kde S (start) a E (z anglického end, konec) jsou skupiny čar a mezer s bitovým vzorem 101, M (z anglického middle, uprostřed) s bitovým vzorem 01010 – se nazývají ochranné vzory. Zbylé znaky L (z anglického left, vlevo) a R (z anglického right, vpravo) zastupují samotné zakódované číslice. Každá číslice je reprezentovaná sedmimístným kódem. Celý kód je tedy reprezentovaný 95 bity. První číslice (L) v kódu je prefix, poslední číslice v kódu (R) je kontrolní číslice, umožňující správně přečíst kód i při chybách při skenování.

Kódy typu EAN

EAN (European Article Number) patří mezi nejrozšířenější čárové kódy, zvláště kód EAN–13. Tyto kódy se používají po celém světě pro označování jednotlivých druhů zboží. Jiné upravené verze těchto kódů umí uchovávat například kódy knih (ISBN) nebo kódy časopisů a jiných tiskovin (ISSN).

Mezi základní EAN kódy patří EAN–2, EAN–5, EAN–8, EAN–13. Z EAN–13 kódu lze zjistit například zemi původu výrobce, způsob užití. Pro užití na menší zboží se používají spíše kódy EAN–8.

EAN–13

Na obrázku 2.2 je příklad kódu EAN–13. Celý kód EAN–13 kóduje třináct číslic a ty jsou rozděleny do čtyř skupin [1].

- Systémové číslice – počáteční dvě až tři číslice se obvykle využívají pro identifikaci země, kde je zaregistrovaný výrobce (to nemusí znamenat to samé jako země původu výrobku).
- Kód výrobce – v závislosti na systémovém kódu se skládá ze čtyř nebo pěti číslic
- Kód výrobku – skládá se z pěti číslic
- Kontrolní číslice – jedná se o tzv. samo–detekující kód (vypočítává se pomocí funkce modulo 10)

– Nejprve se vypočítá suma číslic na lichých pozicích
– Dále suma číslic na sudých pozicích vynásobená třemi
– Předchozí dva výsledky se sečtou
– Součet se zaokrouhlí na desítky nahoru
– Konečně kontrolní číslice je absolutní hodnotou rozdílu předchozích dvou výsledků

Obrázek 2.2: Příklad kódu EAN–13

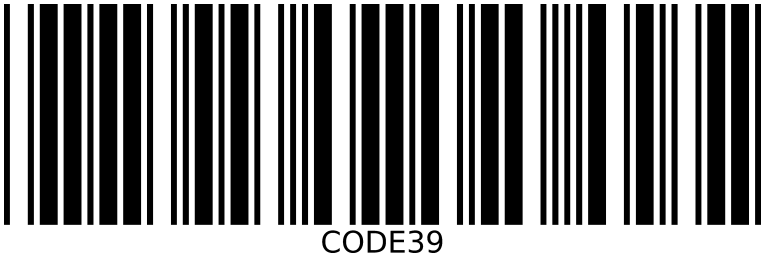
Kód 3/9

Kód 3/9 (také znám jako „USS Code 39“, „Code 3/9“, „Code 39“, …) je čárový kód s proměnnou délkou [9]. Příklad kódu je vidět na obrázku 2.3. Specifikace definuje 43 znaků, které je možné kódovat, velká písmena (A–Z), číslice (0–9) a užitečné speciální znaky (–, ., $, /, +, %, a mezera). Další znak „*“ je používán jako počáteční a koncový oddělovač. Každý znak je tvořen devíti částmi: pěti čarami a čtyřmi mezerami. Tři z devíti částí v každém znaku jsou široké (binární hodnota 1) a šest zbylých částí je úzkých (binární hodnota 0). Poměr mezi šířkou úzkých a širokých částí bývá volen 1:2 nebo 1:3.

Čárový kód neobsahuje žádný kontrolní součet. Největší nevýhodou tohoto kódu je nízká hustota dat. Pro zakódování je zapotřebí mnohem více místa – větší čárový kód než u jiných čárových kódů. Z toho vyplývá, že se tento kód nehodí pro označování malého zboží. Přesto je Kód 3/9 široce rozšířen a může být dekódován pomocí prakticky všech čteček čárových kódů. Jednou z výhod je, že když není nutné generovat kontrolní data, je možné kódovaní jednoduše integrovat do existujícího tiskového systému přidáním písma čárového kódu do systému nebo do tiskárny a pak už jen posílat holá data v tomto písmu.

Kód 3/9 vyvinuli Dr. David Allais a Ray Stevens z Intermecu (1974). Jejich původní návrh zahrnoval dvě široké čáry a jednu širokou mezeru v každém znaku, výsledkem bylo čtyřicet možných znaků pro zakódování. Použitím jednoho z těchto znaků jako počáteční a koncový oddělovač zbývá 39 znaků, odtud původ názvu tohoto kódu. Interpunkční znaky

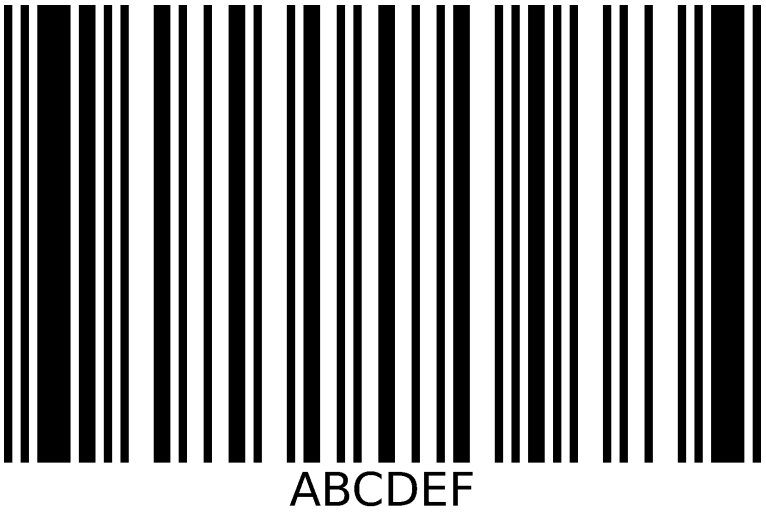
byly později přidány, což se odchýlilo od toho vzoru, rozšířením znakové sady na 43 znaků. Kód 39 byl později standardizovaný jako ANSI MG 10.8 M-1983 a MIL-STD-1189 [5].



Obrázek 2.3: Příklad kódu Code 39

Kód 93

Čárový kód „Kód 93“ (Code 93) byl vyvinutý v roce 1982 v Intermecu jako snaha dosáhnout vyšší datové hustoty a bezpečnosti čárového kódu 3/9 [11]. Pro představu ukázka kódu na obrázku 2.4. Je to alfanumerický kód s variabilní délkou. Byl navržený pro kódování 26 velkých písmen, 10 číslic a 7 speciálních znaků. Každý znak je zakódován pomocí devíti modulů, vždy obsahuje tři čáry a tři mezery (odtud pochází název tohoto kódu). Každá čára/mezera je široká jeden až čtyř násobek atomární šířky.



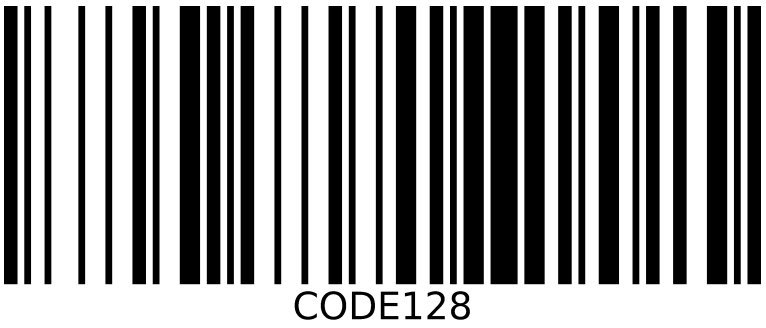
Obrázek 2.4: Příklad kódu Code 93

Rozšířením dosud uvedených 43 znaků o dalších pět speciálních znaků (včetně počátečních a koncových znaků), které můžou být kombinovány s ostatními znaky umožní zakódování znaků z celé kódové sady ASCII (128 znaků).

Kód 128

Kód 128 (nebo také Code 128) je kód s vysokou hustotou dat [8]. Jedna jeho speciální verze nazývaná GS1-128 je používaná celosvětově v oblasti zásilkových služeb. Kódují se s ním alfanumerická nebo pouze numerická data. Může zakódovat všech 128 znaků ASCII. Jako jeden z mála umí rozlišovat velká a malá písmena. Kód má tři kódové sady (A, B, C), na začátku kódovaní se zvolí jedna sada pomocí speciálního znaku. Dále se kóduje pomocí zvolené sady. Sadu je kdykoli možné změnit pomocí dalšího speciálního znaku. První sada umožňuje kódování mimo jiné dolních 32 znaků ASCII (tzv. řídících znaků), druhá ASCII znaky s kódy 32 až 128 a třetí slouží pro kódování dvojciferných čísel 00 až 99. Další znaky, které je možné kódovat (mnohdy stejné pro všechny tři sady) mají většinou speciální význam. Každý znak se skládá ze tří čar a tří mezer určité šířky. Šířka je rovna jedna až čtyřnásobku základní šířky (**X**). Každý znak je zakódován na prostoru o šířce 11 X, s výjimkou ukončovacích znaků, který je dlouhý 13 X.

Předposledním znakem je kontrolní součet. Kontrolní součet se vypočte jako součet násobků jednotlivých kódů vynásobených jejich pozicí a to celé modulo 103. To snižuje možnost výskytu chyby na 1 : 5 000 000.



Obrázek 2.5: Příklad kódu Code 128

Kód 128 byl vyvinut společností Computer Identics (součást Robotic Vission Systems, Inc.) v roce 1980. Na obrázku 2.5 je příklad kódu 128.

2.3 Popis základních typů - 2D čárové kódy

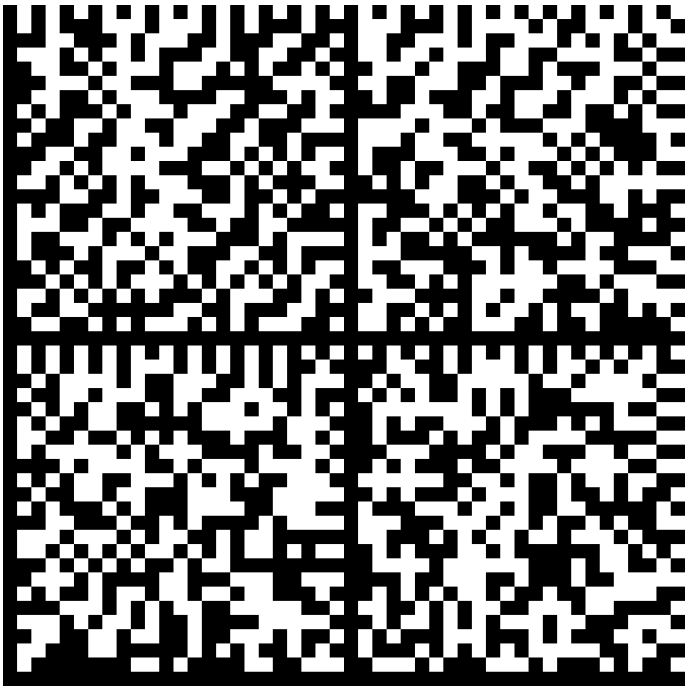
S rostoucí oblibou čárových kódů, díky jejich snadné a rychlé čitelnosti pomocí čtečky, díky jejich přesnosti a vynikajícím funkčním vlastnostem, stoupaly i požadavky na to, co by měly čárové kódy umět. Základními z těchto požadavků bylo uložení více informací, možnost použití větší znakové sady a zároveň menší výsledná velikost. Výsledkem bylo mnoho typů čárových kódů, některé zde už byly uvedeny, nicméně tyto požadavky jsou do jisté míry protichůdné – při zvýšení objemu dat roste velikost kódu. Tento problém se nejprve řešil umístěním několika čárových kódů pod sebe – to byl první krok pro vytvoření dvoudimenzionálních čárových kódů.

Dvoudimenzionální čárové kódy se skládají z černých a bílých buněk uspořádaných ve tvaru obdélníku nebo čtverce. Zavedením další dimenze se zvyšují možnosti kódu, tedy 2D čárové kódy obvykle umožňují zakódovat více dat z větší množiny znaků než 1D čárové kódy

(viz kapitola 2.2). Nyní následuje popis některých nejrozšířenějších dvoudimenzionálních čárových kódů.

Data Matrix

Prvním významným kódem v oblasti dvoudimenzionálních čárových kódu je Data Matrix [3], ukázka na obrázku 2.6. Umožňuje zakódovat text nebo obecná data. Velikost kódovaných dat se obvykle pohybuje od pár bajtů až do dvou kilobajtů. Velikost kódu se mění v závislosti na tom, kolik kódujeme dat a to od rozměrů 8 x 8 až po 144 x 144 (rozměry uvádějí počet sloupců a řádků). Nakonec jsou přidány kódy pro korekci chyb ke zvýšení odolnosti vůči poškození: dokonce i při částečném poškození kódu je možné zakódovaná data přečíst. Data Matrix může uložit až 2335 alfanumerických znaků. Tvar kódu je čtvercový, jednotlivé buňky reprezentují bity. V závislosti na situaci světlé buňky vyjadřují hodnotu 0 a tmavé buňky hodnotu 1 nebo naopak. Každý Data Matrix kód obsahuje dvě přilehlé hranice ve tvaru písmene „L“ (levý a dolní černý okraj). Tato hranice se označuje jako tzv. „tichá zóna“: sama o sobě nenese žádnou informaci, slouží k lokalizaci a zjištění orientace kódu.



Obrázek 2.6: Příklad kódu Data Matrix

Užití

Čárové kódy Data Matrix se používají například jako digitální razítka (STAMPIT Německou poštou). Nejrozšířenější oblastí, kde jsou tyto kódy využívány je označování malých předmětů, díky schopnosti kódu zakódovat padesát znaků do čárového kódu čitelného od velikosti $2 - 3mm^2$ a faktu že kód je možné přečíst pouze s 20% koeficientem kontrastu.

Rozměry vytištěného Data Matrix kódu se můžou velmi lišit, od malých od 300 mikrometrů (vypálené laserem) až po velké o rozměrech 1 metr. Přesnost označovacího a čtecího systému je jediným omezením. EIA (Electronic Industries Alliance) doporučuje používat Data Matrix pro označování malých elektronických komponent [17]. V mnoha průmyslových odvětvích se používají Data Matrix kódy jako součást systému pro sledovaní výroby, zejména v letectví, kde je nutná vysoká kontrola kvality. Data Matrix kódy a doplňující alfanumerická data identifikují detailní informace o komponentách včetně čísla výrobce, čísla komponenty a unikátního sériového čísla.

Standardizace

V dnešní době je standardizace Data Matrix pokryta několika ISO/IEC standardy a je označen jako volné dílo (tzv. public domain) pro mnoho oblastí využití, což znamená, že může být využit bez jakýchkoli licencí nebo poplatků.

- ISO/IEC 16022:2006 – Specifikace čárového kódu Data Matrix
- ISO/IEC 15415–2–D – Standard kvality tisku
- ISO/IEC 15418:2009 – Sémantika datového formátu (identifikátory aplikací GS1 a ASC MH10 datové identifikátory)
- ISO/IEC 15424:2008 – Specifikace datového formátu včetně identifikátorů symbolů (identifikátory pro rozlišení různých typů čárového kódu)
- ISO/IEC 15434:2009 – Syntaxe pro vysokokapacitní ADC média (formát dat přenášených ze skeneru do softwaru, atd.)
- ISO/IEC 15459 – Unikátní identifikátory

Ačkoli je tento standard volný, neexistují žádné volně dostupné dokumenty vysvětlující proces kódování Data Matrix. Dokumentace se dá zakoupit přes webové stránky organizace ISO [2].

QR kód

QR kód je druh 2D čárového kódu vyvinutý Denso Wave (divize společnosti Denso Corporation – v té době) v roce 1994 [10]. Primárním cílem bylo vytvoření kódu, který bude snadno čitelný pomocí čtečky. Na obrázku 2.7 je ukázka QR kódu.

Vlastnosti

Možnost zakódovat velké množství dat QR kód umožňuje zakódovat řádově mnohem větší množství dat než klasické čárové kódy. Krom toho navíc ještě i různé druhy dat jako jsou čísla, písmena, japonská (a některá další asijská) znaková písma a binární data. Konkrétní čísla uvádějící kapacitu QR kódu najdete v tabulce 2.1.

Malá velikost výsledného kódu Zakódovaný QR kód zabírá přibližně 10krát menší plochu než klasický čárový kód při kódování stejného množství dat. V případě potřeby ještě menšího kódu byla vyvinuta speciální verze QR kódu, tzv. Mikro QR kód.

Datová kapacita QR kódu		
Čísla	maximálně 7089 znaků	Každé tři číslice zakódovány do 10–ti bitů
Znaky, čísla a některé další znaky	maximálně 4296 znaků	Každé dva znaky jsou zakódovány do 11–ti bitů
Binární data	maximálně 2953 bajtů	V každém bajtu 8 bitů
Japonská znaková písma	maximálně 1817 znaků	Každý znak zakódovaný jako do 13–ti bitů

Tabulka 2.1: Datová kapacita QR kódu

Odolnost vůči poškození QR kód používá samoopravné kódy, díky nimž je při poškození možné kód správně dekódovat. Samoopravným kódem používaným k těmto účelům je Reed–Solomon [7]. Samoopravné kódy fungují až do 30% poškození. QR definuje čtyři úrovně zabezpečení samoopravnými kódy, každá úroveň se dokáže vypořádat s jinak velkým poškozením. Platí, že čím větší schopnost opravy, tím menší kapacita kódu.

Jednotlivé úrovně opravných kódů:

- úroveň L: opraví až 7% chyb.
- úroveň M: opraví až 15% chyb.
- úroveň Q: opraví až 25% chyb.
- úroveň H: opraví až 30% chyb.

Obrázek 2.7: Příklad QR kódu

Rychlá čitelnost z libovolného směru QR kód je možné vysokou rychlostí číst z libovolného úhlu a to díky napevno definovaným vzorům v každém QR kódu. Při čtení jsou tyto poziční vzory detekovány a na základě nich určit natočení v prostoru. Ve třech vrcholech kódu jsou umístěny soustředné čtyřúhelníky, ve čtvrtém vrcholu pak značka ve tvaru menšího čtyřúhelníku.

Verze QR kódu QR kódy mohou mít různé velikosti. Jednotlivé velikosti jsou definovány jako verze QR kódu. Celkem je definováno 40 verzí. Verze 1 je matice o velikosti 21 x 21 bodů. V každé další verzi se velikost zvětšuje o čtyři body až do verze 40, což je matice o rozměrech 177 x 177 bodů.

Standardizace

Kompletní specifikaci QR kódu pokrývají následující dokumenty:

- JIS X 0510 (JIS – Japanese Industrial Standards, tedy japonské průmyslové standardy)
- ISO/IEC 18004:2000
- ISO/IEC 18004:2006

Specifikace na aplikační vrstvě není jednotná, existují různé implementace pro kódování různých typů dat jako jsou URL, elektronické vizitky a další.

Licence

QR kód je možné používat bez omezení bez jakékoli licence, byl jasně definován jako ISO standard. Společnost Denso Wave sice vlastní patentová práva, ale rozhodla se je neuplatňovat [10].

PDF417

PDF417 je více-řádkový lineární čárový kód s proměnnou délkou, vysokou datovou kapacitou a samoopravnými schopnostmi [6]. Používá se v mnoha oblastech, především v dopravě a řízení zásob. PDF je zkratka pro Portable Data File. PDF417 byl vyvinut doktorem Ynjiun P. Wang v Symbol Tenchologies v roce 1991. Kód je specifikován ISO standardem 15438. Příklad kódu PDF417 je na obrázku 2.8. Umožňuje zakódovat více než 1100 bajtů, 1800 alfanumerických znaků nebo 2710 číslic. Dokáže kódovat celou ASCII tabulku znaků a 8-bitová binární data. Rozměry kódu jsou 3 až 90 řádků a 90 až 538 atomických šířek (šířka nejužší čáry v kódu) na šířku.

Vlastnosti

Kromě vlastností typických pro dvoudimenzionální čárové kódy patří mezi schopnosti PDF417 tyto:

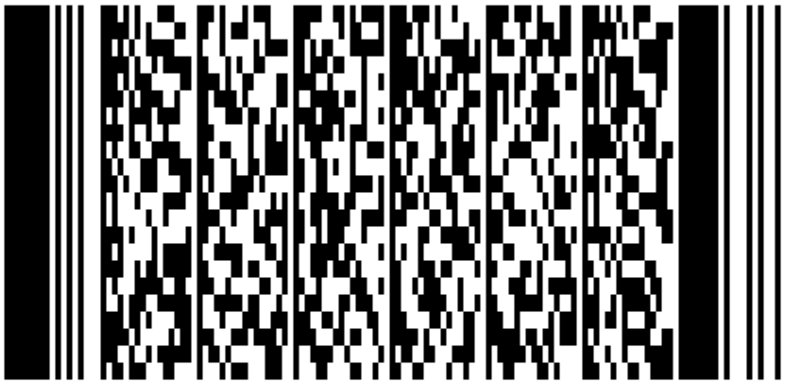
- Propojování. Kódy PDF417 můžou být propojeny s jinými kódy které jsou pak skenovány v sekvenci, to umožňuje zakódovat více dat.

- Uživatelsky specifikovatelné rozměry. Uživatel může zvolit jak široká bude nejužší svislá čára (osa X) a jak vysoké budou řádky (osa Y).
- Volné dílo (public domain), kdokoli může systém implementovat bez jakýchkoli licencí.

Formát

Kód PDF417 se skládá z jednotlivých řádků, každý řádek potom obsahuje:

- Tichou oblast, to je minimální bílá oblast před samotným kódem.



Obrázek 2.8: Příklad kódu PDF417

- Počáteční vzor, který identifikuje formát kódu PDF417. Každý typ čárového kódu má unikátní počáteční a ukončující vzor.
- Levostranný identifikátor kódového slova obsahuje informace o řádku jako je číslo řádku a použitý koeficient pro samoopravný kód v tomto řádku.
  - 1 až 30 datových kódových slov. Kódová slova jsou skupiny čar a mezer reprezentující jednu nebo více číslic, písmen nebo jiných symbolů.
  - Všechny řádky mají stejný počet kódových slov.
  - Každé kódové slovo se skládá ze čtyř čar a čtyř mezer (odtud pochází číslo 4 v názvu kódu).
  - Celková šířka kódového slova je 17krát šířka atomární šířky (odtud pochází číslo 17 v názvu kódu).
  - Každé kódové slovo začíná čárou a končí mezerou.
  - Specifikace umožňuje volit z 929 kódových slov, 900 pro data a 29 pro různé speciální účely.
  - Každé kódové slovo je vykresleno pomocí jedné ze tří odlišných množin kódových slov nazývaných clustery:
    * Cluster je tedy vzor čar a mezer pro všechny z 929 kódových slov.
    * Žádná kombinace čar a mezer se neopakuje mezi clustery.
    * Číslo řádku rozlišuje, který cluster se použije.

    - Pro všechny kódová slova v řádku se používá stejný cluster.
    - Účelem clusterů je možnost určit, ze kterého řádku kódové slova je. To umožňuje číst kód i z jiných úhlů.

- Pravostranné kódové slovo s dalšími informacemi o řádku.
- Ukončovací vzor.
- Tichá zóna.

2.4 Volba typu čárového kódu pro tento systém

Bylo zde uvedeno velké množství různých čárových kódů a to jsou jen ty rozšířenější. Původním účelem čárových kódů (myšleny jsou 1D čárové kódy) bylo označování zboží/součástek při zvyšování efektivity práce v továrnách a velkých společnostech. Dnešní nároky mají mnohem širší záběr. Díky tomu bylo možné si pro tuto práci vybírat z relativně velkého množství možností. Základními požadavky pro tuto práci jsou:

- Rychlost – je nutné kód přečíst pohotově, aby to uživatele nezdržovalo
- Spolehlivost – i za zhoršených podmínek musí být kód přečten správně
- Kapacita – kód musí umožňovat zakódovat dostatečné množství dat
- Velikost – kód se má zobrazovat na displeji zařízení proto musí mít vhodné rozměry a relativně omezenou velikost

Určitě by se našli další čárové kódy, ale tyto základní zde uvedené postačí pro volbu, který čárový kód využiji pro tuto práci. Z požadavků jasně vyplývá, že mnohem vhodnější budou 2D čárové kódy – mají mnohem větší kapacitu, čtvercový/obdélníkový tvar s relativně malou velikostí (vhodné pro zobrazení na mobilním zařízení). Mezi 2D kódy byly favority Data Matrix [3] a QR kód [10]. Nakonec jsem zvolil QR kód, jelikož pro Data Matrix kód není volně dostupný žádný dokument popisující jeho kódování (přesto, že kód sám o sobě je tzv. Public Domain).

Kapitola 3

Bezpečnost

Podstatnou součástí navrhovaného systému je i bezpečnost. Je třeba zabránit, aby citlivá data padla do nesprávných rukou. V této kapitole se tedy budou brát v úvahu všechny možnosti, jak by bylo možné systém napadnout a navrhují se taková opatření, která mají za úkol útoky jednotlivých typů znemožnit.

3.1 Autorizace

Asi nejpodstatnější částí celého procesu zajištění bezpečnosti je autorizace uživatele [14]. Autorizací všeobecně rozumíme oprávnění, schválení nebo pověření. Proces autorizace označuje získání přístupu k informacím, funkcím a dalším objektům. Skládá z:

- Autentizace subjektu
- Vyhledání v seznamu oprávněných subjektů, jejich rolí a práv.
- Udělení oprávnění nebo odepření přístupu.

Seznam oprávnění je v informatice realizován přidělením oprávnění například pro práci s konkrétními soubory nebo adresáři, pro provedení operace či přístupu k jiným prostředkům v počítači. Autorizaci provádí operační systém nebo specializovaný software na základě seznamů pro řízení přístupu.

Autentizace

Autentizace je proces ověření proklamované identity subjektu [14]. V informatice autentizace značí proces ověření identity uživatele služeb nebo původce zprávy. Pro zjištění identity se používají tyto základní metody:

- na základě informace známé pouze konkrétnímu uživateli (zná správnou kombinaci uživatelského označení a hesla nebo PIN)
- na základě vlastnictví konkrétní věci (USB klíč, smart card, soukromý klíč apod.)
- podle toho, čím uživatel je (ověřitelné biometrické vlastnosti – otisk prstu, snímek oční duhovky či sítnice, DNA apod.)
- autentizace na základě dovednosti (uživatel umí správně odpovědět na náhodně vygenerovaný kontrolní dotaz)

Na základě úspěšnosti autentizace se rozhoduje, zda bude autorizace uživateli udělena nebo zamítnuta.

Nedostatky stávajících způsobů autorizace finančních transakcí

Správa osobních účtů a realizace finančních transakcí prostřednictvím různých systémů elektronického bankovnictví, které v dnešní době patří ke každodenním činnostem mnoha z nás. Jakákoliv manipulace s finančními prostředky (a zejména jedná–li se o vyšší obnosy) je už dlouhou dobu považována za velmi citlivou operaci – přitahuje totiž nežádoucí pozornost jednotlivců či organizovaných skupin hledajících možnosti, jak se snadno a rychle obohatit na úkor ostatních.

Ať je současný model v jakémkoliv směru dokonalý, postavíme-li do role útočníka samotného obchodníka, zjistíme, že téměř žádný z používaných bezpečnostních prvků mu samostatně nemůže zabránit zneužít svého postavení a podvést zákazníka. Postavení obchodníka je výjimečné tím, že pro platby poskytuje „důvěryhodný terminál“. Stačí mu tedy jen například podstrčit falešný displej zobrazující sumu rozdílnou od té, která je právě odečítána ze zákazníkova účtu. Zákazník na místě nemá žádnou šanci takovou transakci před provedením potvrdit či zastavit, obrana je možná až v případě, kdy se vyskytne větší množství stížností na jednoho obchodníka.

Obecně platí, že terminál je pod výhradní kontrolou obchodníka, platební karta pod kontrolou banky, avšak zákazník nemá k dispozici žádnou technologii, která by mu umožnila ověřit, že obchodník zadal skutečně správnou sumu (tj. tu, která se zobrazuje na displeji terminálu).

Podobně je na tom i uživatel přistupující ke svému účtu přes systémy elektronického bankovnictví. Zde je jako bezpečná autorizační metoda mnohdy používána čipová karta s uloženými soukromými klíči a certifikáty veřejných klíčů. Pokud však útočník získá kontrolu nad celým počítačem a odposlechne PIN, může této čipové kartě neomezeně zasílat příkazy k autorizaci nejrůznějších transakcí.

Toto neuspokojivé postavení klienta je hlavním důvodem pro reálnou potřebu levného a jednoduchého zařízení komunikujícího s platební kartou (či jiným tokenem), které by bylo výhradně pod kontrolou zákazníka, a umožňovalo by mu plnou kontrolu nad zpracováváním transakcí – ideálně zobrazením dat, např. částky a čísla cílového účtu, které jsou posílány čipové kartě k podpisu.

3.2 Šifrování

Šifrování a dešifrování dat tvoří menší, ale podstatnou část této práce. Zajišťuje bezpečnost dat, se kterými se pracuje. Vzhledem k tomu, že dochází k aplikaci několika různých šifer, je třeba tuto tématiku trochu více rozebrat.

Proč šifrovat?

Nejdůležitější důvody pro šifrování jsou čtyři [14]. Dobrá šifra by měla zajišťovat:

- důvěrnost – skutečný obsah dat zůstane skrytý nepovolaným osobám
- autentizaci – odesílatel dat není zaměnitelný
- nepopiratelnost – odesílatel dat je jednoznačně identifikovatelný

- integritu – obsah zprávy nelze upravovat

Zajímavostí je, že žádná ze soudobých šifer neumí zajistit splnění všech těchto bodů. Proto se používají kombinace, kdy se šifrovaný text znovu šifruje jiným způsobem.

Jak šifrovat?

V dnešní době je výkon počítačů příliš vysoký, než aby bylo snadné vytvořit vlastní šifru tak, aby byla dostatečně bezpečná. Zároveň nelze využít princip *security through obscurity* (bezpečnost přes nejasnost), kdy se spoléhá na utajení šifrovacího algoritmu před třetí stranou.

Zbývá tedy pouze zvolit již existující algoritmus. Ty se dělí na dvě odlišné skupiny:

- symetrické šifrování
- asymetrické šifrování

Symetrické šifrování

Nejznámějším a nejstarším typem šifrování je šifrování symetrické. Funguje tak, že odesílatel i příjemce zprávy se předem dohodnou na společném tajném klíči, který odesílatel použije na šifrování dat. Po přijetí šifrovaného textu stejným klíčem data příjemce i dešifruje.

Výhodou tohoto principu je jeho relativní jednoduchost a rychlost. Obsah zprávy zůstává důvěrný a podle zvoleného algoritmu je zachována i integrita. Pomocí symetrické šifry se však nedá zajistit autentizace ani nepopiratelnost.

V současnosti se nejčastěji lze setkat s algoritmy IDEA, DES, blowfish, TEA a hlavně DES a AES.

AES

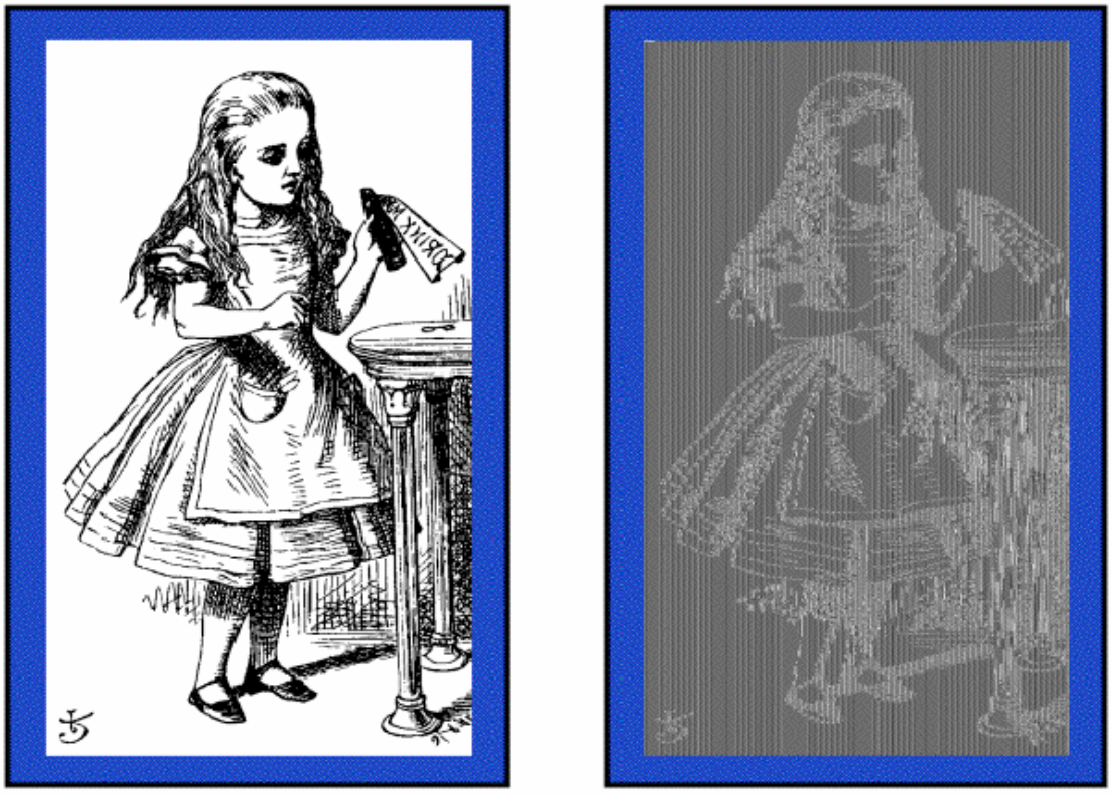
Tento algoritmus bude v systému použit a zaslouží si proto bližší popis. Zkratka AES znamená *Advanced Encryption Standard* (pokročilý šifrovací standard). Algoritmus byl vytvořen jako náhrada staršího DES a snaží se odstranit jeho nedostatky. Ty spočívají hlavně ke končící životnosti DES. Výkon dnešních strojů již totiž umožňuje prolomit šifru hrubou silou – tedy postupným zkoušením všech možných klíčů. Životnost algoritmu AES byla v roce 2002, kdy byl schválen jako federální standard USA, odhadována minimálně na 20 let.

Jedním z důvodů je zvětšení délky klíče z 56 bitů (DES) na 128, 192, popř. 256 bitů, čímž znemožňuje útok hrubou silou pomocí současné techniky.

Stejně jako DES se jedná o blokovou šifru: Data jsou šifrována po blocích o 128 bitech, které jsou poté zřetězeny. Algoritmus při šifrování prochází sekvencí 10, 12 nebo 14 smyček (podle délky klíče), ve kterých je prováděna permutace a substituce jednotlivých bitů bloku podle klíče. Operace zároveň obsahují nelineární krok, který značně zvyšuje odolnost šifry proti útokům.

Samotná implementace musí ještě řešit problém se zřetězením bloků. Naivní implementace blokové šifry by spočívala v aplikaci šifry na jednotlivé bloky tak, jak jdou za sebou. Tomuto postupu se říká ECB (*Electronic Codebook* – elektronická kniha kódů) a jeho použití se nedoporučuje. Jeho výsledkem totiž je, že stejné bloky otevřeného textu budou zašifrovány vždy stejně. Vzhledem k poměrně malé velikosti jednotlivých bloků to

může znamenat, že ve výsledné zprávě budou patrná stejná schémata jako v původním dokumentu (viz obrázek 3.1). Útočník může navíc s bloky manipulovat (duplikovat, přeházet, odstranit…).



Obrázek 3.1: Viditelná opakující se schémata při použití ECB u blokové šifry [12]

V praxi se tedy používají takové algoritmy, které toto nebezpečí odstraňují. Nejjednodušším z nich je CBC – *Cipher Block Chaining* (řetězení šifrovaných bloků). Postup funguje tak, že před zašifrováním se nad odpovídajícím blokem otevřeného textu provede operace XOR s předcházejícím blokem již zašifrovaného textu. To znamená, že jednotlivé bloky jsou na sobě závislé. Aby bylo možné dešifrovat konkrétní blok, je třeba nejprve dešifrovat i všechny předchozí.

Protože není čím XORovat první blok, vytvoří se ještě jeden „nultý“ (*inicializační vektor*) a použije se k dešifrování prvního bloku. Do zprávy se už nepřidává.

Rychlost šifrování a jeho bezpečnost dělá z AES vhodného kandidáta pro použití v této práci.

Asymetrické šifrování

V porovnání se symetrickým je asymetrické šifrování velmi mladé. Hlavním rozdílem je skutečnost, že k přenosu zprávy jsou třeba dva rozdílné klíče. Tyto klíče se liší svým určením, kde jeden z nich slouží k zašifrování zprávy a druhý k jejímu dešifrování.

K asymetrickým šifrám patří například Knapsack, DSA a asi nejznámější RSA.

Posílání zpráv pak může probíhat následujícím způsobem: Odesílatel požádá adresáta, aby vygeneroval sadu klíčů a zaslal mu klíč potřebný k zašifrování zprávy. Ten může být

poslán nešifrován a přečíst si jej tedy může kdokoliv (proto se mu říká *veřejný klíč*). Odesílatel poté zprávu zašifruje veřejným klíčem a pošle ji adresátovi. Zpráva je pak dešifrována druhým (*soukromým*) klíčem. Protože je veřejný klíč znám, může kdokoliv vytvořit vlastní šifrovanou zprávu, ale pouze majitel soukromého klíče ji dokáže dešifrovat.

V tomto případě je zajištěna pouze důvěrnost zprávy, nelze však zaručit ani autentizaci, ani nepopiratelnost, ani integritu.

Druhou alternativou posílání zpráv je tzv. *elektronický podpis*: Odesílatel oba klíče vygeneruje sám. Pomocí soukromého klíče zprávu zašifruje a pošle ji adresátovi. Zároveň také zveřejní svůj veřejný klíč. Příjemce zprávy pomocí veřejného klíče zprávu dešifruje. Data tedy dokáže pomocí veřejného klíče dešifrovat každý, ale pouze majitel soukromého klíče dokáže data zašifrovat.

Pokud je potvrzen původ veřejného klíče (k tomu slouží certifikáty a certifikační autority), pak tento princip zajistí autentizaci, nepopiratelnost a se správným algoritmem i integritu dat. Na druhou stranu však nezajistí jejich důvěrnost.

Pro zajištění všech čtyř bodů se používá kombinace předchozích dvou postupů: Data jsou nejprve zašifrována soukromým klíčem odesílatele a poté znovu veřejným klíčem příjemce. Po přijetí příjemce „rozlepí obálku“ pomocí svého soukromého klíče a poté „přečte podpis“ pomocí veřejného klíče odesílatele.

Nevýhodou asymetrického šifrování je však výpočetní náročnost vlastního procesu šifrování.

Nutno podotknout, že bezpečnost jakéhokoliv šifrování – ať už symetrického nebo asymetrického – je podmíněna neprozrazením klíče (kromě veřejného).

Certifikáty a certifikační autority

Během šifrování pomocí asymetrických šifer stále vzniká problém s napadnutelností systému, pokud se útočníkovi podaří podstrčit vlastní veřejný klíč v okamžiku zahájení šifrované komunikace. V takovém případě se útočník vydává za příjemce nebo odesílatele šifrovaných zpráv a druhá strana komunikace nedokáže tuto záměnu rozpoznat.

Aby se předešlo tomuto typu útoku, zavádí se takzvané *certifikační autority* [13] – důvěryhodné třetí strany, které podepíší veřejný klíč generovaný odesílatelem vlastním soukromým klíčem. Spolu s klíčem odesílatele jsou podepsány i jeho identifikační údaje. Takto vygenerovaná zpráva se nazývá *certifikát* a jeho úkolem je delegování zodpovědnosti za pravost zpráv certifikační autoritě.

Napadnutelnost komunikace se tedy přesouvá na komunikaci mezi uživatelem a certifikační autoritou v okamžiku udělení certifikátu. Ta však nemusí probíhat zdaleka tak často a v případě vysoké potřeby bezpečnosti ji lze zajistit i neelektronickou cestou.

Kapitola 4

Návrh systému

V této kapitole bude řešen samotný cíl práce, čímž je návrh a implementace systému pro zabezpečený přenos dat pomocí čárových kódů. V podstatě jde o vytvoření základního systému přenosu dat založeného na vizuálním přenosu dat, kdy jsou na jedné straně (vysílači) zobrazena data zakódovaná do čárového kódu a na straně druhé (přijímači) je pomocí snímacího zařízení (kamery) čárový kód nasnímán a rozkódován. Tento základní systém pro přenos dat je pak možné rozšířit o různé prvky zabezpečení. Tím je zajištěno, že budou data doručena správně a nebude je možné zneužít neoprávněnou osobou. Jednotlivé prvky zabezpečení jsou popsány jako koncepty řešení zabezpečení. Pro konkrétní využití systému v praxi v určité konkrétní aplikaci jsou zvoleny požadované koncepty. Takto je možné pomocí dopředu navržených a otestovaných konceptů poskládat požadovaný systém pro konkrétní aplikaci.

Systém bude aplikovatelný do různých oblastí, kde je nutné zadávat citlivá data. Může sloužit jako bezpečná schránka pro uschování citlivých dat k řadě různých systému. Krom úschovy ale hlavně zajistí bezpečný přenos dat do druhého systému.

Způsobů pro přenos dat je mnoho, dílčím cílem práce je porovnat vizuální přenos s jinými a najít vhodné oblasti pro uplatnění tohoto způsobu přenosu dat.

4.1 Jádro systému

Jádrem systému je implementace základního přenosu dat vizuální cestou. V základu se jedná o jednosměrnou komunikaci, bude tedy potřeba vytvořit přijímač a vysílač.

Mimo funkcí přijímače a vysílače budou mít výsledné aplikace další funkce, které budou navrhnuty a popsané dále. Aplikace určená pro uživatele systému bude dále nazývaná klient a aplikace určená především k funkci terminálu pro napojení na další systémy bude dále nazývaná server.

Vysílač

Vysílačem je myšlen mobilní telefon či jiné zařízení vybavené aplikací, která přenášená data zakóduje do QR čárového kódu (volba typu čárového kódu viz kapitola 2.4). Výsledný kód poté zobrazí na displeji zařízení.

Přijímač

Funkcí přijímače je získat data z vysílače. Může se jednat o libovolné zařízení vybavené skenerem QR čárových kódů nebo o mobilní telefon/počítač vybavený kamerou a příslušným softwarem. V případě použití již hotového skeneru QR čárových kódů je první krok funkce přijímače přeskočen. V další části budeme předpokládat využití kamery pro snímání. QR kód je nasnímán kamerou z displeje vysílače. Poté je ve snímku lokalizován samotný kód, který je pak dekódován – tím jsou získaná data odeslaná vysílačem.

Klient

Práce se má zabývat bezpečným přenosem dat, krom toho je ale nutné zabezpečit data v mobilním telefonu pro případ ztráty nebo odcizení. Toto zabezpečení bude realizováno zašifrováním veškerých dat v mobilním zařízení pomocí symetrické šifry AES (viz kapitola 3.2). K rozšifrování dojde při spuštění aplikace zadáním přístupového hesla. Úroveň zabezpečení je poté už dána hlavně délkou hesla a tím, jestli a do jaké míry je uživatel schopen zajistit neprozrazení svého hesla.

Další ochranné prvky jsou pak již součástí rozšiřujících konceptů, základní verze předpokládá zakódovávání čistého textu do čárového kódu.

Kromě šifrování má klient také zajišťovat přístup a práci s citlivými daty pro různé jiné systémy. Musí být možné přidávat, upravovat a hlavně používat množství tzv. profilů. Profilem se rozumí všechny informace k jednomu jinému systému.

Server

Část systému (označovaná jako server) slouží především jako napojení na jiné systémy. Server se skládá z obecné části a pak části specifické pro každý cizí systém, se kterým má být použitý. Základní část zajišťuje vizuální komunikaci s případnými rozšířeními v podobě volby některých konceptů pro zabezpečení apod. Implementace části pro komunikaci s cizím systémem záleží na konkrétním systému.

4.2 Rozšiřující koncepty

Konceptem je myšleno navržené řešení nějaké obecné situace, v této práci se jedná hlavně o koncepty týkající se bezpečnosti. Koncepty se poté dají podle požadavků kladených na výsledný systém skládat a různě kombinovat. Tím lze dosáhnout dostatečné bezpečnosti, ovšem ne na úkor jiných parametrů, mezi které patří hlavně jednoduchá použitelnost systému.

Potvrzení identity

V případě potvrzení identity jde o dokázání, že uživatel je opravdu tím, za koho se vydává. Toho lze docílit pomocí digitálního podpisu (viz kapitola 3.2). Uživatel si nechá od nějaké autority, tedy třetí strany, vygenerovat soukromý a veřejný klíč. Zvolená autorita musí být důvěryhodná pro všechny zúčastněné. Veškerá komunikace je poté před odesláním elektronicky podepsaná uživatelovým soukromým klíčem. Tím je zajištěno, že zpráva pochází opravdu od správného uživatele a také, že zprávu nikdo nemodifikoval.

Zabezpečení relace

Zabezpečením relace je myšlen způsob zabezpečení aktuální operace, tedy komunikace mezi dvěma zařízeními. Jde o zajištění toho, že i v případě, kdy se útočníkovi podaří prolomit ochranu při jedné operaci, nebude moci využít získaných informací v jiný čas (při jiné operaci). Toho lze docílit různými způsoby. Každý způsob zajišťuje jinou míry bezpečí. Platí, že čím větší míra bezpečnosti, tím více kroků musí uživatel provést při operaci a tím delší dobu operace probíhá. Proto je vhodné zvolit způsob, který bude kompromisem mezi bezpečností a uživatelskou přívětivostí.

První možností jak zabezpečit relaci je vytvoření časově se měnícího kódu. Ke zprávě je připojeno časové razítko vytvoření zprávy. Výsledná data jsou před započetím komunikace elektronicky podepsána (viz kapitola 4.2). Tím je omezena platnost dat na určitý časový interval. Interval platnosti časového razítka zvolíme podle úrovně zabezpečení. Pokud chceme nastavit interval vypršení platnosti kódu na čas $= t$, pak je vhodné implementovat tuto funkci tak, že interval změny razítka nastavíme $2t$ a necháme server kladně přijímat poslední dva časové kódy. Tím předejdeme situaci, kdy oprávněný klient vygeneruje kód těsně před změnou časového razítka, ale použít se ho pokusí až po změně razítka. Přestože v tomto příkladu nedošlo ke zneužití, dojde k odmítnutí kódu.

Další uvedenou možností pro zabezpečení relace je obdoba prvního způsobu. Místo časového razítka se však ke zprávě připojí krátký kód zadaný uživatelem (PIN) podle pokynů ze serveru. Server při čekání na klienta vždy vygeneruje náhodně určitou krátkou sekvenci znaků, které pak uživatel před začátkem relace zadá do klientské aplikace. Při komunikaci server porovnává tento kód. Platnost kódu je omezena jen pro jednu relaci.

4.3 Definice komunikačního standardu

Komunikační standard definuje způsob komunikace mezi klientem a serverem. Každá aplikace či zařízení splňující tyto pravidla může tento systém využívat.

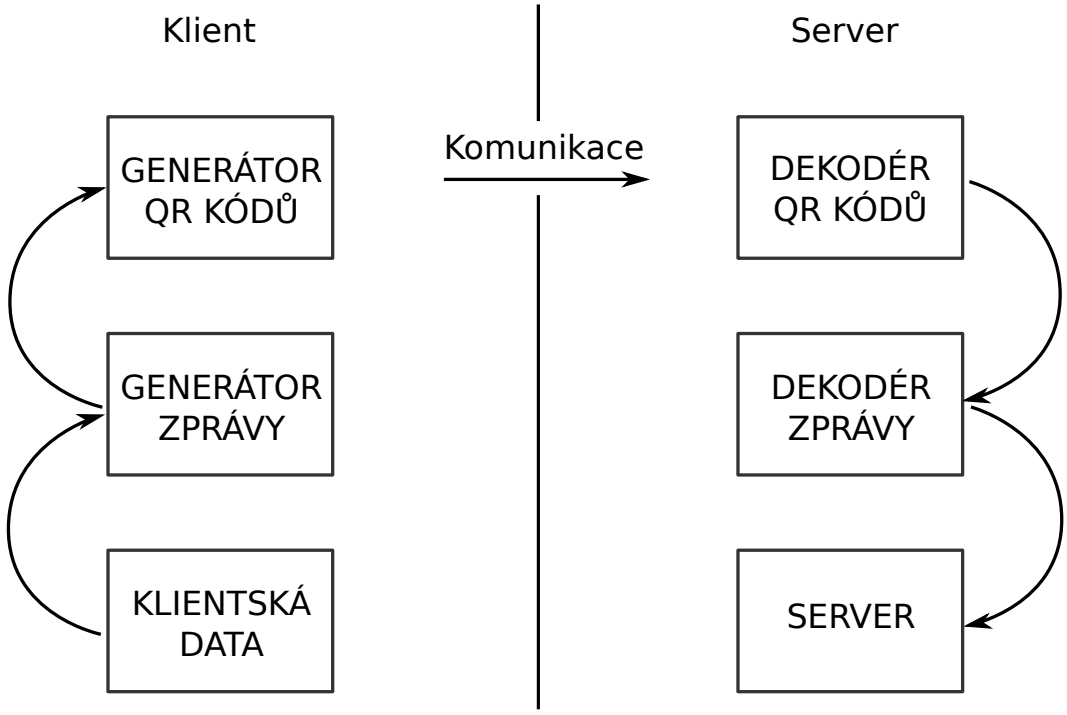
Komunikace

Na obrázku 4.1 je blokově znázorněna komunikace mezi klientem a serverem. Standard definuje přenosové médium, tedy QR čárové kódy a formát přenášených dat. Konkrétní způsob generování QR kódů apod. už nejsou jeho součástí.

Komunikace probíhá vizuálním přenosem informací v podobě QR čárových kódů. Před přenosem je nutné data vhodně upravit. Blok „Klientská data“ zahrnuje klientskou aplikaci pro výběr dat, která budou přenášena. Následuje výběr dat a jejich formátování, přičemž jsou dále doplněna o další informace. K tomu dojde v bloku označeném jako „Generátor zprávy“. Tento blok bude podrobněji popsán v následující kapitole – formát zprávy 4.3.

Upravená data jsou poté zakódována do QR čárového kódu. K tomu dojde v bloku „Generátor QR kódů“, po přenosu dat je QR kód rozkódován v bloku „Dekodér QR kódů“. Obsah těchto bloků není součástí specifikace standardu. Umožňuje to použití systému v širším spektru aplikací. Je možné použít jak softwarovou knihovnu, tak i libovolný jiný systém jako hardwarový modul.

Po rozkódování zprávy z QR kódu je na přenesené data v bloku „Dekodér zprávy“ aplikovaný stejný postup jako při generování zprávy, jen v opačném pořadí. Tím jsou získány



Obrázek 4.1: Schéma komunikace mezi klientem a serverem

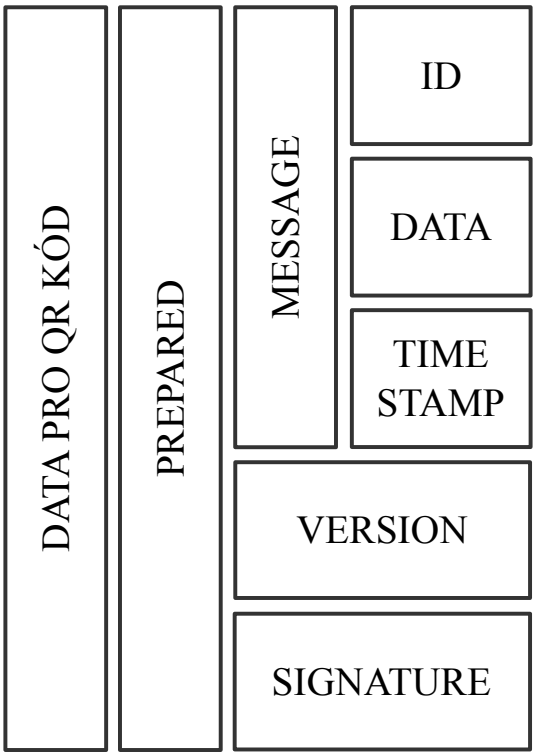
původní přenášená data a navíc je zajištěno, že data byla odeslána autorizovaným uživatelem. Konečná data může serverová aplikace zpracovat v bloku „Server“ podle konkrétní aplikace.

Formát zprávy

Nejprve budou popsány jednotlivé bloky zprávy podle schématu na obrázku 4.2. Poté bude následovat postup pro postupné sestavení zprávy.

- Blok „ID“ označuje identifikaci klienta, přičemž každému klientu je přidělen jedinečný identifikátor skládající se ze sekvence sedmi znaků a číslic. Tím dostáváme 36^7 kombinací, tzn. přes 78 miliard kombinací.
- V bloku „DATA“ jsou přenášena libovolná (volitelná) data specifická pro konkrétní aplikaci.
- Blok „TIMESTAMP“ slouží k přenosu časového razítka vytvoření zprávy – časové razítko je ve formátu *Unix time*, což určuje počet sekund uplynulých od 1.1.1970 západoevropského času (UTC+0). Druhou možností je krátký kód pro identifikaci konkretní komunikace – tento kód je nutné uživatelsky zadat před každým generování zprávy podle pokynů z druhé strany, tedy serveru. Tento kód se náhodně generuje z množiny číslic a má délku 4, to znamená 10 tisíc kombinací.

- Pro vygenerování bloku „MESSAGE“ jsou použity všechny předchozí bloky. Předchozí bloky se sestaví do textové podoby ve formátu XML, označení jednotlivých bloků v XML je specifikované v tabulce 4.1.
- Za pomoci bloku „MESSAGE“ je vygenerován další blok, a to „SIGNATURE“. Blok představuje hlavní zabezpečení standardu. Generování bloku se provede výpočtem hašovací funkce z dat v předchozím uvedeném bloku. Pro výpočet hašovací funkce je použit algoritmus *SHA-1*. Nakonec je výsledek této funkce elektronicky podepsán, tzn. zašifrován pomocí asymetrické šifry *RSA*, a to privátním klíčem klienta (viz kapitola 3.2). Pro *RSA* jsou použity klíče o délce 1024 bitů.



Obrázek 4.2: Formát zprávy

- V bloku „VERSION“ je specifikována verze zprávy. Verze je definována pomocí jednoho znaku z množiny písmen a číslic (tedy 36 kombinací). To umožňuje případnou rozšiřitelnost standardu. V této práci jsou specifikovány dvě verze pro generování zprávy, a to následující:
  - Verze *A* – identifikovaná znakem *A*. Tato verze v bloku „TIMESTAMP“ přenáší časové razítko.
  - Verze *B* – identifikovaná znakem *B*. Blok „TIMESTAMP“ je nevyužit.
- Blok „PREPARED“ je sestaven z bloků „MESSAGE“, „VERSION“ a „SIGNATURE“. Sestavení proběhne generováním XML, specifikace je opět v tabulce 4.1.

- Blok „DATA PRO QR KÓD“ je konečná modifikace dat do podoby vhodné pro kódování do QR čárového kódu. Cílem je dosáhnout toho, aby QR kód byl „nejmenší“, tedy byl zakódován co nejefektivněji. QR čárový kód nejefektivněji kóduje číslice. Proto jsou data z předchozího bloku převedena na sekvenci čísel. Převod na sekvenci čísel se provádí tak, že každý znak je převeden na hexadecimální číslo podle *ASCII* tabulky. Výsledné hexadecimální číslo je převedeno na číslo o základu deset.

Blok	**Označení v XML**
ID	$< i >$
DATA	$< d >$
TIMESTAMP	$< t >$
VERSION	$< v >$
MESSAGE	$< m >$
SIGNATURE	$< s >$

Tabulka 4.1: Označení komponent zprávy v XML

Pro zakódování/rozkódování zprávy jsou potřeba další doplňující data. Těmito daty jsou hlavně klíče pro asymetrickou kryptografii. Jsou potřeba rozdílná data pro klientskou a serverovou aplikaci. V klientské aplikaci se jedná o privátní klíč vygenerovaný při registraci klienta. V serverové aplikaci je uložen odpovídající veřejný klíč identifikovaný podle klientského identifikačního čísla.

Verze protokolu

Protokol pro generování a přenos zprav navržený v této práci umožňuje využít různé funkce zabezpečení relace, ty jsou specifikované verzí protokolu. V rámci práce jsou navrženy dvě verze. V případě rozšíření požadavků je díky způsobu řešení verzí možné rozšířit systém o další verze. Stávající verze jsou:

- Verze *A* – jak již bylo popsáno v kapitole 4.3 – při použití této verze se kóduje do zprávy časové razítko, které identifikuje přesný čas vytvoření zprávy. Pomocí informace z časového razítka je zajištěno zabezpečení relace (jedna z možností jak relaci zabezpečit).

- Verze *B* – na rozdíl od předchozí uvedené verze používá odlišný způsob zabezpečení relace. Relace je zabezpečena pomocí krátkého kódu generovaného druhou stranou (serverovou aplikací), tento kód se nazývá *PIN*. Do generátoru zprávy je nutné *PIN* přenést, toho je docíleno pomocí klientské aplikace, která umožní *PIN* uživateli zadat pomocí uživatelského rozhraní. Uživatel tedy musí *PIN* přečíst z obrazovky serverové aplikace a sám jej vložit.

Generování zprávy

Formát zprávy popsaný v kapitole 4.3 definuje pouze strukturu, nyní bude popsaný postup vytváření kompletní zprávy. Generování bude v některých částech závislé na zvolené verzi zprávy. Hned prvním rozdílem jsou odlišné požadavky na vstupní data. Vstupní data pro jednotlivé verze generátoru zpráv jsou následující:

- Verze *A* – ID, data, soukromý šifrovací klíč.
- Verze *B* – ID, data, *PIN*.

Obě verze na vstupu očekávají položku ID, což je identifikátor uživatele. Tímto identifikátorem je rozpoznán uživatel na straně serverové aplikace. Další položkou, kterou očekávají obě verze jsou data, tuto položku není nutné vyplňovat, je volitelná. Data je možné využít pro přenos dat specifických pro různé užití systému.

Další položky jsou už závislé na verzi. Pro verzi *A* je to soukromý šifrovací klíč. Ten je vygenerován při registraci nebo obdržení přístupových údajů do jiného systému. Při použití v asymetrické kryptografii tvoří soukromý klíč dvojici s veřejným klíčem (viz kapitola 3.2). Při generování zprávy je nutný právě soukromý klíč, jelikož je využíván pro elektronický podpis. Tak je zajištěno, že serverová aplikace může ověřit, zda nedošlo k podvrhnutí zprávy a zda je tedy uživatel oprávněný vykonat požadovanou činnost. Verze *B* vyžaduje pro každé generování nový krátký kód, již dříve označený jako *PIN*.

Vstupní data jsou poté postupně transformována. V případě generování zprávy verze *A* je nejprve vygenerováno časové razítko vytvoření zprávy – jeho formát je popsaný v kapitole 4.3. Poté je z časového razítka, dat a ID sestaven jeden blok pojmenovaný *Message* – formát tohoto bloku je popsaný v kapitole 4.3. Pro zprávu ve verzi *B* je ponechána položka s časovým razítkem prázdná, jinak je postup totožný.

Následující krok probíhá pouze při generování zprávy verze *B*. Tímto blokem je *Signature* – jedná se o již zmiňovaný elektronický podpis, ten je vygenerován pomocí bloku *Message* a soukromého šifrovacího klíče. Nejprve se pro data v bloku *Message* vypočítá hašovací funkce a poté je výsledek této funkce elektronicky podepsán zašifrováním soukromým klíčem. Algoritmy a jejich parametry jsou podrobněji popsány v kapitole 4.3. Pro zprávu verze *A* je tento blok ponechán prázdný.

Nyní jsou opět některé bloky spojeny pro převod do podoby vhodné pro kódování do QR čárového kódu. Blok obsahující výsledné spojené bloky se nazývá *Prepared*. Sestavení tohoto bloku je podrobně popsáno v kapitole 4.3. Posledním krokem je pouze transformace bloku *Prepared* – tato transformace je taktéž popsána v uvedené kapitole.

Dekódování zprávy

Dekódování zprávy probíhá v podstatě stejně jako generování, pouze v opačném pořadí. Jako vstup dekodéru zprávy je výstup z dekodéru QR čárového kódu. Tyto data jsou transformována do bloku označovaného jako *Prepared*, postup je inverzní ke generování bloku *Prepared* (viz kapitola 4.3). Tím jsou získána data ve formátu XML. Z XML dat je možné přímo získat verzi zprávy. Podle verze se budou některé následující kroky lišit.

V případě, že zpráva je verze *A*, je nejprve nutné ověřit elektronický podpis zprávy, tím se předejde případnému podvržení zprávy. Ověření podpisu probíhá tak, že je nejprve pro blok dat označený jako *Message*, který je přímo získán z XML dat, vypočítána hašovací funkce. Poté je nutné blok dat označený jako *Signature* rozšifrovat pomocí veřejného klíče

uživatele. Veřejné klíče jednotlivých uživatelů jsou uloženy někde na straně serveru. Pro dohledání příslušného veřejného klíče je nejprve nutné znát ID uživatele. To je získáno rozložením bloku dat označeného jako *Message* (tento blok obsahuje další bloky ve formátu XML). Blok *Message* je tedy rozložen na bloky *ID*, *Data* a *Timestamp*. Podle získaného ID uživatele je tedy dohledán veřejný šifrovací klíč uživatele a tímto klíčem je poté rozšifrován blok *Signature*. Výsledek rozšifrování je porovnán s už dříve spočítaným výsledkem hašovací funkce a pokud jsou naprosto stejné, je potvrzeno, že zpráva byla vytvořena autorizovaným uživatelem a už s ní nebylo později nijak manipulováno. Pokud tedy zpráva pochází od autorizovaného uživatele, následuje dekódování zprávy verze *A*. V tomto případě se jedná o zabezpečení relace pomocí časového razítka. Na základě toho je tedy nutné ověřit platnost zprávy. Nejprve je potřeba vygenerovat vlastní časové razítko pro aktuální čas, které je poté porovnáno s časovým razítkem dekódované zprávy. Teoreticky je možné v dekodéru zpráv zvolit čas, po který bude zpráva platná libovolně podle specifických požadavků konkrétního způsobu užití systému. Pro většinu aplikací je vhodné volit čas na jednu minutu. Zpráva je tedy platná, pokud je maximálně o jednu minutu starší než je aktuální čas. Pro správnou funkci systému je nutné, aby obě aplikace (jak serverová, tak klientská) měly správně nastavený čas. V případě serverové aplikace je v zájmu provozovatele, aby byl čas přesný a aby se zabránilo užití systému neautorizovanému uživateli. V případě klientské aplikace je zase v zájmu uživatele, aby QR čárový kód, který je jeho aplikací vygenerován, neplatil delší dobu než je doporučeno pro případ krádeže kódu.

Postup pro zprávu verze *B* je následující – blok *Message* je rozšifrován symetrickou šifrou podle klíče označeného jako *PIN*. Ten je nutné už dopředu vygenerovat a poté si ho pro tento krok pamatovat (generování položky *PIN* viz kapitola 4.3). Pokud se podaří zprávu rozšifrovat, pak už jsou přímo dostupná všechny data (položky *ID* a *Data*). V opačném případě nebyla zpráva zašifrována pomocí aktuálně platného *PIN* kódu a není tedy platná.

Pokud jsou tedy potvrzeny všechny ochranné prvky zprávy, může serverová aplikace zpracovat případné volitelně přenášená data a podle toho provést příslušné akce. Pro základní identifikaci postačuje identifikátor uživatele, podle kterého serverová aplikace pozná, který uživatel chce provést danou funkci.

Limity

Při návrhu formátu zprávy bylo nutné brát v úvahu různá omezení. Hlavním omezením byla velikost zprávy, kterou je nutno zachovat co nejmenší, a to hned z několika důvodů. Jedním z důvodů je, že zpráva bude přenášena jako QR čárový kód, který má sice oproti jiným čárovým kódům kapacitu velkou, ale přesto ne dostatečnou pro přenos většího objemu dat. V případě použití kryptografie velikost dat rychle narůstá. Kdyby se navíc například přenášely se zprávou i certifikáty pro ověřování elektronických podpisů, jak se to dělá standardně při ověřování elektronických podpisů (například v internetových prohlížečích), velikost dat by značně převyšovala kapacitu QR kódu. Navíc je nutné brát v úvahu to, že QR čárový kód bude načítán z displeje elektronického zařízení, které má proti standardnímu použití, kdy je čárový kód vytištěn na papíře či jiném povrchu, omezené rozměry a rozlišení. Kvalita zobrazení na displeji elektronického zařízení je navíc velice citlivá na světelné podmínky v místě použití – přímé světlo způsobuje odlesky a velmi snižuje kontrast obrazu.

Dalším aspektem, který je nutné brát na zřetel, je schopnost snímacího zařízení přečíst čárový kód i při příznivých podmínkách, tedy v případě, že snímání není zatíženo problémy popsanými v předchozím odstavci. Teoretická kapacita QR čárového kódu je více než 7000

číslic (viz kapitola 2.3). Přesto je pro snadné použití i s nepříliš kvalitním fotoaparátem, kterým jsou zpravidla vybaveny mobilní zařízení, vhodné kódovat maximálně okolo 1000 číslic do jedné přenášené zprávy.

Z těchto důvodů byly tedy navrhnuty různé metody, jak velikost zprávy zmenšit (například při využívání již zmíněného elektronického podpisu). Do zprávy se kóduje pouze samotný elektronický podpis, tedy zašifrovaný výsledek hašovací funkce z podepisovaných dat. Veřejný klíč nutný pro potvrzení autentičnosti zprávy je celou dobu uchováván na straně serverové aplikace. Pro snadné dekódování je ve zprávě mimo jiné přenášen identifikátor uživatele, podle kterého je příslušný veřejný klíč dohledán.

4.4 Požadavky na technologie

Proč vizuální přenos informace

Využití čárových kódů v této práci se odchyluje od jejich typického využití. Při jejich návrhu pravděpodobně nikdo nepředpokládal využití pro přenos většího objemu dat. S příchodem 2D čárových kódů už je to možné alespoň technologicky.

V této práci je potřeba přenášet poměrně velké množství informací. I když originálních dat je celkem málo a bylo by je možné přenášet velmi jednoduše, díky nutnosti zabezpečit přenos různými ochrannými prvky – jako je šifrování – objem dat hodně naroste.

Kromě čárových kódů se samozřejmě nabízela celá řada jiných přenosových technologií. Předtím než však objasním volbu tohoto prostředku, uvádím krátký přehled jednotlivých přenosových technologií a jejich výhody a nevýhody použití v této práci.

Wi–Fi První technologií v dnešní době velmi rozšířenou pro přenos informací je Wi–Fi, jedná se o bezdrátový přenos informací rádiovým signálem o frekvenci 2,4 GHz.

Výhody:

- Rychlost/přenosové pásmo – umožňuje přenést obrovské množství informací ve velmi krátké době
- Obousměrná komunikace

Nevýhody:

- Cena – náklady na hardware jsou nejvyšší ze všech technologií zde uvedených
- Spotřeba – energetické nároky jsou – vzhledem k tomu, ze systém má být používán s mobilním zařízením, – poměrmě velké
- Zbytečně velký dosah signálu – možnost odposlechnutí
- Jen v dražších telefonech, hlavně smartphone

Bluetooth Relativně podobná technologie jako Wi–Fi, využívá také rádiového signálu o frekvenci 2,4 GHz. Na rozdíl od Wi–Fi je určena spíše pro malá zařízení jako je právě mobilní telefon.

Výhody:

- Rychlost/přenosové pásmo – umožňuje přenést obrovské množství informací ve velmi krátké době
- Obousměrná komunikace
- Většina dnešních telefonů je vybavena Bluetooth

Nevýhody:

- Spotřeba – energetické nároky jsou poměrně velké vzhledem k tomu, že systém má být používán s mobilním zařízením
- Navazování spojení – před použitím je nutná interakce uživatele

Infračervené záření Přenos informace na krátkou vzdálenost s přímou viditelností mezi vysílačem a přijímačem pomocí infračerveného světla.

Výhody:

- Rychlost
- Obousměrná komunikace

Nevýhody:

- V dnešní době už se příliš nepoužívá, proto chybí podpora v moderních zařízeních

Čárové kódy Jde o vizuální přenos informace pomocí grafického kódu zobrazeného na jedné straně a čtecího zařízení na straně druhé, přičemž toto čtecí zařízení je použito k načtení obrázku kódu, jeho rozpoznání a rozkódování.

Výhody:

- Využívá digitálního fotoaparátu, kterým je v dnešní době vybaven téměř každý mobilní telefon

Nevýhody:

- Jednosměrný přenos dat
- Omezená kapacita dat na jeden kód

Vhodných způsobů přenosu informací je jistě více, avšak přenos vizuální cestou pomocí čárových kódů není dosud v praxi odzkoušen. Toto využití je z hlediska použití čárových kódů nové. Hlavním cílem této práce je odhalit případné nedostatky a navrhnout řešení, jak tento způsob komunikace použít v praxi.

Nároky na mobilní zařízení

Pro přenos informací pomocí čárových kódu stačí pouze zobrazit kód na displeji jednoho zařízení a na druhém zařízení kód pomocí kamery vyfotit a načíst. Přesto má tento způsob přenosu informací specifické hardwarové požadavky.

Nejdůležitějšími součástmi mobilního zařízení potřebnými pro přenos pomocí čárových kódů jsou tedy displej a kamera. Hlavním parametrem displeje je rozlišení (tedy počet bodu na výšku a šířku) a zároveň jeho rozměry. V závislosti na parametrech displeje je možné pomocí kamery načíst kód. Pro kameru jsou taktéž nejdůležitější parametry rozlišení, ale také parametry týkající se schopnosti snímat i při zhoršených světelných podmínkách.

Obecně platí, že čím kvalitnější snímací zařízení, tedy kamera, tím může být méně kvalitní zobrazovací zařízení, tedy displej. Vzhledem k tomu, že systém má být použitelný v praxi, je nutné najít určitý kompromis v poměru kvality, který by splňovala většina běžně dostupných mobilních zařízení. Konkrétní hodnoty vyplynou z testování.

Parametry hardware se ale nedají testovat bez přesných požadavků, kterými jsou hlavně kvantita dat – tedy množství dat, které bude tímto způsobem přenášeno. Čím méně dat, tím je samozřejmě možné použít méně kvalitní hardware. V tomto systému ovšem množství dat záleží na použitých konceptech – u některých dochází k velkému nárůstu množství přenášených dat hlavně z důvodu šifrování.

Pokud je potřeba přenášet takové množství dat, které by nebylo možné přenášet na většině dostupných zařízení, a tedy by nebylo možné systém jednoduše zavést do praxe, lze data přenést v sekvenci. Při přenosu dat v sekvenci se postupně na jedné straně zobrazují kódy na displeji a na druhé straně jsou postupně načítány.

Kapitola 5

Realizace systému

V této kapitole bude popsáno, co vše bylo implementováno v rámci této práce a jaké nástroje k tomu byly využity. Hlavním cílem práce bylo navrhnout platformu pro realizaci zabezpečeného přenosu dat. Proto vyvinutá ukázková aplikace slouží hlavně k demonstraci možností systému. Ukázková aplikace se skládá ze tří hlavních součástí: klientské aplikace, serverové aplikace a knihovny implementující generování a dekódování přenášené zprávy se všemi prvkami zabezpečení popsanými v kapitole 4.3.

5.1 Knihovna *Cryptor*

Knihovna *Cryptor*, jež vznikla jako hlavní programový výstup této práce, implementuje základní funkcionalitu systému pro zabezpečený přenos informací vizuální cestou s využitím QR čárových kódů. Knihovna tvoří jen část celkového systému – konkrétně část nazvanou *Generátor a dekodér zprávy*, která je popsána v kapitole 4.3. Knihovna tedy slouží ke generování a dekódování dat přenášených pomocí QR čárového kódu. Formát zprávy je nadefinovaný v kapitole 4.3.

Knihovna je implementována v jazyce *Java*. Díky tomu, že je tato komponenta vyvinuta jako programová knihovna, je ji možné využít při implementaci libovolné klientské nebo serverové aplikace. Komponenta je distribuována jako *Java* knihovní balík `JAR`. V případě požadavku použití na platformě, která nepodporuje platformu *Java*, je možné implementovat podobnou knihovnu. Použitím knihovny je zajištěna požadovaná bezpečnost – knihovna implementuje dvě metody zabezpečení relace a využívá k tomu různé druhy kryptografie.

Struktura

Knihovna *Cryptor* je rozdělena do dvou balíků. První balík označený jako `cryptor.crypt` shromažďuje třídy zabývající se logikou ohledně zabezpečení a šifrování. Druhý balík označený jako `cryptor.message` na rozdíl od předchozí obsahuje třídy zajišťující samotné generování/dekódování zprávy.

Knihovní balík „cryptor.crypt“

Tento balík jak již bylo uvedeno shromažďuje třídy, které mají na starost zabezpečení a šifrování. V balíku jsou obsaženy tři třídy. První třídou je `AES`, která je pojmenovaná podle symetrické šifry *AES*, jejíž funkcionalitu implementuje. Třída tedy umožňuje šifrovat a

dešifrovat text pomocí šifry *AES* – popis šifry viz kapitola 3.2. Další třídou je `RSA`, která je pojmenovaná taktéž podle šifry, tentokrát asymetrické šifry *RSA*. Tato třída umožňuje rovněž šifrovat a dešifrovat pomocí dané šifry. Navíc oproti předchozí třídě umožňuje tato ještě vygenerování a poté správu asymetrických klíčů nutných pro práci s šifrou *RSA* (více o asymetrickém šifrování v kapitole 3.2). Poslední třídou obsaženou v tomto balíku je pomocná třída pojmenovaná `HexCodec`. Třída slouží k převodu mezi kódováním řetězců – konkrétně mezi standardním binárním kódováním a mezi hexadecimálním kódováním.

Knihovní balík „cryptor.message“

Druhý knihovní balík, jak již název napovídá, obsahuje třídy potřebné pro generování a dekódování zprávy. První třídou je `MessageAdapter`, což je třída obsahující hlavní logiku generování a dekódování zprávy. Další skupina tříd slouží k uchovávání struktury zprávy v jednotlivých fázích (konkrétně se jedná o třídy `Message`, `Prepared` a `Result`). Blokové schéma zprávy je znázorněno na obrázku 4.2.

Obrázek 5.1: Zadávání hesla při spouštění klientské aplikace

Do poslední skupiny patří třídy jednotlivých výjimek, které mohou nastat při generování nebo dekódování zprávy. První třídou z této skupiny je třída `NoSuchVersionException`,

která je vyvolána v případě, že je generovaná či dekódovaná zpráva verze, kterou tato knihovna nepodporuje. Další výjimka je vyvolána v případě, že se dekóduje zpráva zabezpečená pomocí časového razítka a stáří zprávy přesáhlo nastavený limit. Tato výjimka je implementována třídou `OverageMessageException`. Výjimka implementovaná třídou `FakedMessageException` je vyvolána v případě, že se dekóduje zpráva zabezpečená pomocí elektronického podpisu a podpis nesouhlasí – tedy jedná se o podvrženou zprávu. Poslední třída `BadPinException` implementuje výjimku vyvolanou v případě, že se dekóduje zpráva zabezpečená pomocí *PIN* kódu a zároveň pokud tento kód není validní (tedy pokud byl kód špatně zadán případně použit podruhé).

5.2 Klient

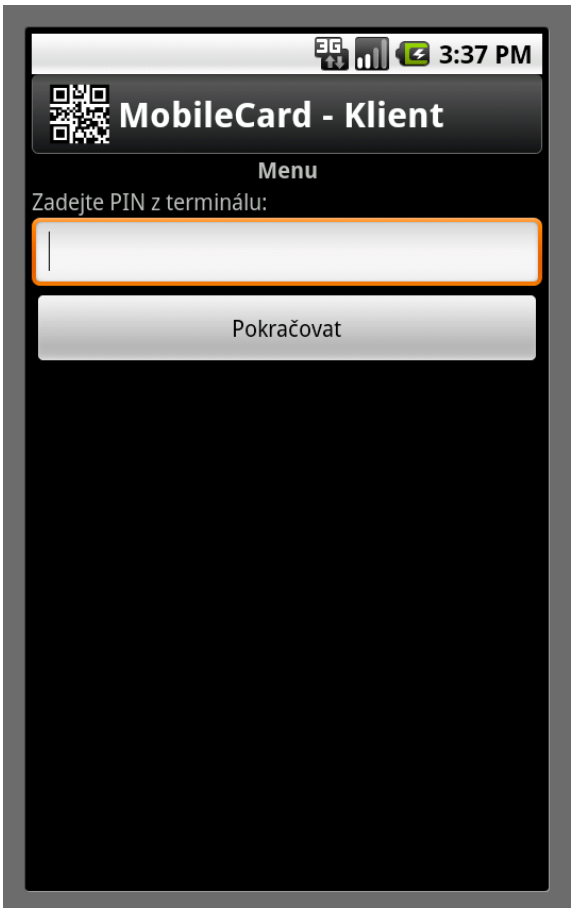
Klient (neboli klientská aplikace) slouží k bezpečnému uschování různých přístupových profilů k jednotlivým systémům. Umožňuje přístup k těmto profilům a na základě uložených dat generuje zabezpečené zprávy ve formátu QR čárových kódů podle standardu popsaného v kapitole 4.3.

Obrázek 5.2: Menu s volbami jednotlivých profilů v klientské aplikaci

Pro bezpečné uchování dat je vhodné veškerá uživatelská data v aplikaci při ukládání do paměti zařízení šifrovat. Přístup do aplikace by měl být možný pouze po zadání správného hesla, teprve poté by mělo dojít k rozšifrování potřebných dat.

Pro provoz klientské aplikace je nejvhodnější mobilní telefon nebo osobní asistent. Použitím zařízení tohoto typu je zajištěna dostupnost pro široké spektrum uživatelů. Tato zařízení jsou navíc vybavena veškerým potřebným hardwarem.

Funkce klientské aplikace se tedy skládá z několika základních činností. V prvé řadě uchovává klientská data v zašifrované podobě v paměti zařízení. Pak umožňuje správu přístupových profilů (přístupovým profilem je myšlena skupina dat, která umožňuje uživateli pomocí komunikace mezi klientskou a serverovou aplikací provést požadovanou operaci). Při použití tohoto systému je zajištěna bezpečnost procesu komunikace. V prvé řadě je zajištěno, že požadovanou operaci může provést pouze uživatel, který na to má právo. Krom bezpečné úschovy dat aplikace dále umožňuje po výběru profilu vygenerovat zabezpečenou zprávu pomocí generátoru zprávy popsaného v kapitole 5.1.



Obrázek 5.3: Zadávání PINu ze serverové aplikace do klientské

Účelem této práce je v prvé řadě návrh a otestování možností bezpečného přenosu informace pomocí QR čárových kódů. Proto klientská demonstrační aplikace vyvinutá v rámci práce nesplňuje úplně všechny popisované vlastnosti – slouží hlavně k předvedení funkčnosti systému. Aplikace je vyvinutá pro platformu *Android* a napsaná v programovacím jazyce

Java. Mezi implementované funkce patří v prvé řadě generování zpráv podle standardu (viz kapitola 4.3). Další implementovanou funkcí je výběr profilů. Profily ovšem není možné uživatelsky spravovat a přidávat případně další. Další funkcí, která v demonstrační aplikaci není implementována, je šifrování dat v paměti zařízení.

Ovládání

V této kapitole bude přiblíženo ovládání demonstrační klientské aplikace. Demonstrační aplikace slouží především k nastínění funkce podobné aplikace pro použití mezi reálnými uživateli, proto nejsou některé funkce implementované plně. Co se týče funkcionality, je upřednostněna plná integrace generátoru zpráv, jelikož se jedná o klíčovou funkci celého systému.

Obrázek 5.4: Zobrazení vygenerovaného QR čárového kódu v klientská aplikaci

- Ihned po spuštění klientské aplikace je potřeba zadat vstupní přístupové heslo. V demonstrační aplikaci je tato funkce pouze z důvodu představy funkčnosti reálné aplikace, není tedy implementována část realizující samotné šifrování dat. V reálné aplikaci by se ve chvíli zadání hesla provedlo dešifrování uživatelských dat z paměti zařízení. Pro představu lze najít ukázku na obrázku 5.1.

- Po správném zadání hesla se zobrazí další obrazovka aplikace (viz obrázek 5.2). Na této obrazovce je zobrazen seznam jednotlivých přístupových profilů. Také by zde mělo být možné dále s profily pracovat, upravovat je, případně přidávat další. Pro jednoduchou orientaci jsou zde zobrazeny krátké názvy jednotlivých profilů a volitelně uveden i krátký popis.
- Poté, co uživatel zvolí jeden z přístupových profilů, dojde ke generování zprávy. Tato operace je závislá na verzi zprávy, která se má generovat. Verze zprávy je obsažena v informacích uložených v rámci přístupového profilu.
  - Při volbě přístupového profilu, který je určen pro generování zprávy verze *A*, tedy zprávy zabezpečené pomocí časového razítka, není nutná žádná uživatelská interakce – veškeré operace proběhnou na pozadí aplikace. Výsledkem je vygenerovaná zpráva.
  - Naopak při generování zprávy verze *B* (tedy zprávy primárně zabezpečené pomocí kódu *PIN* vyměněného mezi serverovou a klientskou aplikaci pro každou relaci) je nezbytné zmíněný *PIN* kód uživatelsky zadat. K tomuto účelu klientská aplikace nabídne klientovi formulář pro zadání (viz obrázek 5.3). Po zadání *PIN* kódu jsou opět na pozadí aplikace provedeny potřebné výpočty a na základě nich vygenerována zpráva.
- Vygenerovanou zprávu je následně potřeba zobrazit. K tomuto účelu slouží další obrazovka klientské aplikace, na které je zpráva zobrazena ve formátu QR čárového kódu. Ukázka zobrazení vygenerované zprávy se nachází na obrázku 5.4.

5.3 Server

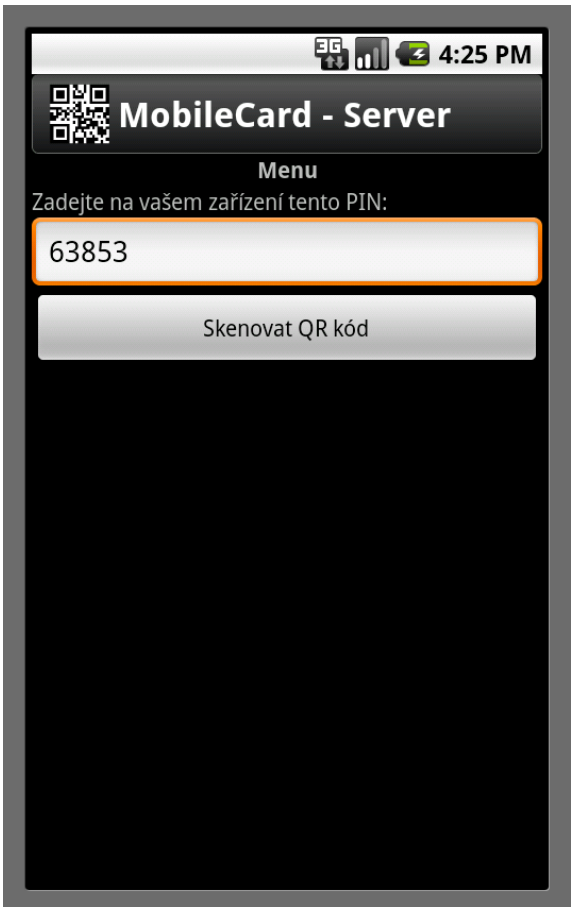
Server (neboli serverová aplikace) plní funkci dekodéru zprávy od klienta. Na základě dekódované zprávy potom provádí požadovanou činnost. K tomu využívá další systémy. Příkladem činnosti serverové aplikace a systému, který serverová aplikace využívá pro vykonání této činnosti, může být například otevírání vstupních dveří. Při otevírání dveří může vzniknout potřeba se připojit hned na několik systému, přičemž hned prvním bude elektronický zámek, který umožní samotné otevření dveří. Dalším může být například určitá služba, která podle údajů v databázi zjistí, zda daný uživatel má přístup do příslušných dveří, atd. Konkrétní oblasti, kde se dá zabezpečený přenos informací pomocí QR čárových kódů využít v praxi, budou popsány v kapitole 6.

Serverová aplikace se skládá z několika bloků. Jádrem aplikace je dekodér zprávy (viz kapitola 5.1). Předtím, než je možné dekódovat data, probíhá jejich získání, načtení a první typ rozkódování (rozkódování QR čárového kódu). Pro získání dat libovolnou metodou snímání QR čárových kódů může být použit CCD snímač a poté může být kód softwarově detekován v obrazu a dekódován, případně lze použít hardwarovou čtečku, která jako výstup poskytuje přímo data z čárového kódu. Dalším blokem je napojení na ovládaný systém. V praxi se předpokládá možnost záměny jednotlivých bloků beze změny funkcionality systému jako takového.

V rámci této práce vznikla demonstrační serverová aplikace, která je implementovaná na platformě pro mobilní zařízení Android (více viz kapitola 5.4). Implementačním jazykem

je *Java*. Z pohledu jednotlivých bloků, ze kterých se má každá serverová aplikace skládat, implementuje vyvinutá demonstrační aplikace plně blok pro dekódování zprávy (viz kapitola 5.1).

Blok pro získání dat od klientské aplikace je implementován s využitím knihovny pro rozkódování QR čárového kódu – použitá knihovna se jmenuje `ZXing` (více viz kapitola 5.4). Pro samotné získání snímku čárového kódu je využito vestavěné kamery v mobilním zařízení, na kterém má být demonstrační aplikace provozována.



Obrázek 5.5: Vygenerovaný PIN v serverové aplikaci

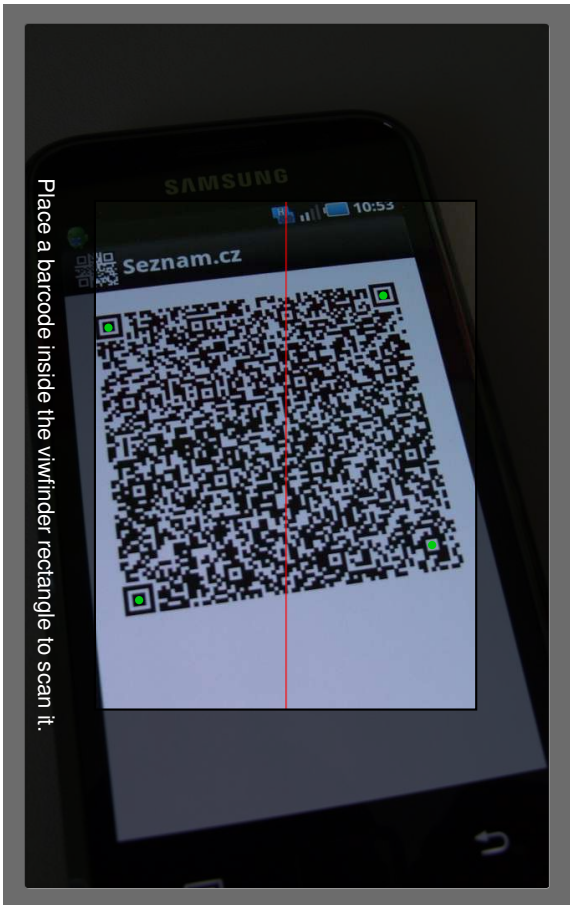
Poslední součástí každé serverové aplikace by měl být blok pro napojení na daný systém a jeho ovládání. Pro účely demonstrační aplikace je implementován jen zjednodušený blok, který se nenapojuje na žádný další systém. Slouží pouze ke strukturovanému zobrazení dekódovaných dat.

Ovládání

Nyní bude podrobněji popsána demonstrační serverová aplikace včetně uživatelského rozhraní. Na rozdíl od aplikací pro reálný provoz v praxi umožňuje demonstrační serverová aplikace přijímat všechny verze zpráv. Následuje popis jednotlivých kroků při přijímání zprávy od klientské aplikace:

- Po spuštění aplikace se zobrazí jednoduchá obrazovka se dvěmi položkami – textovým polem zobrazujícím náhodně vygenerovaný *PIN* kód a tlačítkem s textem „Skenovat QR kód“ pro přepnutí aplikace do módu skenování QR čárového kódu obsahujícího přenášenou zprávu. Ukázka přímo z demonstrační aplikace je vidět na obrázku 5.5.

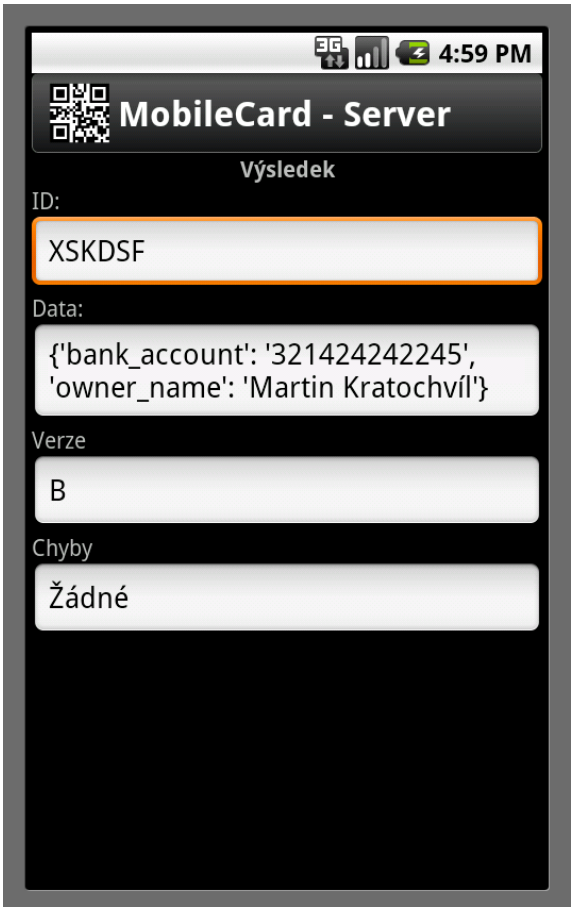
  – Pro přijmutí zprávy ve verzi *A*, tedy zprávy zabezpečené pomocí časového razítka, není položka s *PIN* kódem nijak využita, stačí tedy pouze stisknout tlačítko pro skenování QR čárového kódu.
  – Pro zprávu verze *B* je naopak položka s *PIN* kódem velmi důležitá, představuje hlavní bezpečnostní prvek pro přenos zprávy. Zobrazenou hodnotu v této položce je nutné zobrazit uživateli klientské aplikace, aby ji mohl zadat do svého zařízení ještě před vygenerováním zprávy. Poté opět stejně jak v případě přijmutí zprávy verze *A* stačí pouze stisknout tlačítko pro skenování QR čárového kódu. Hodnota pro položku s *PIN* kódem je po každém rozkódování zprávy vygenerována znovu.



Obrázek 5.6: Skenování QR kódu v serverové aplikaci

- Ukázka skenování QR čárového kódu v demonstrační aplikaci je vidět na obrázku 5.6. Pokud se aplikaci podaří detekovat a naskenovat QR čárový kód, přepne se automaticky na další obrazovku – obrazovku se zobrazením výsledků.

- Na poslední obrazovce demonstrační aplikace jsou zobrazeny výsledky dekódování přenášené zprávy. V případě, že dekódování proběhne bez chyb, jsou zde vidět hodnoty jednotlivých položek. Pokud ale dojde při dekódování k potížím, které se nejčastěji vyskytují při ověřování platnosti zprávy, je na této obrazovce popsána příslušná chyba. Ukázka, jak může vypadat takové zobrazení výsledků, je zobrazena na obrázku 5.7.



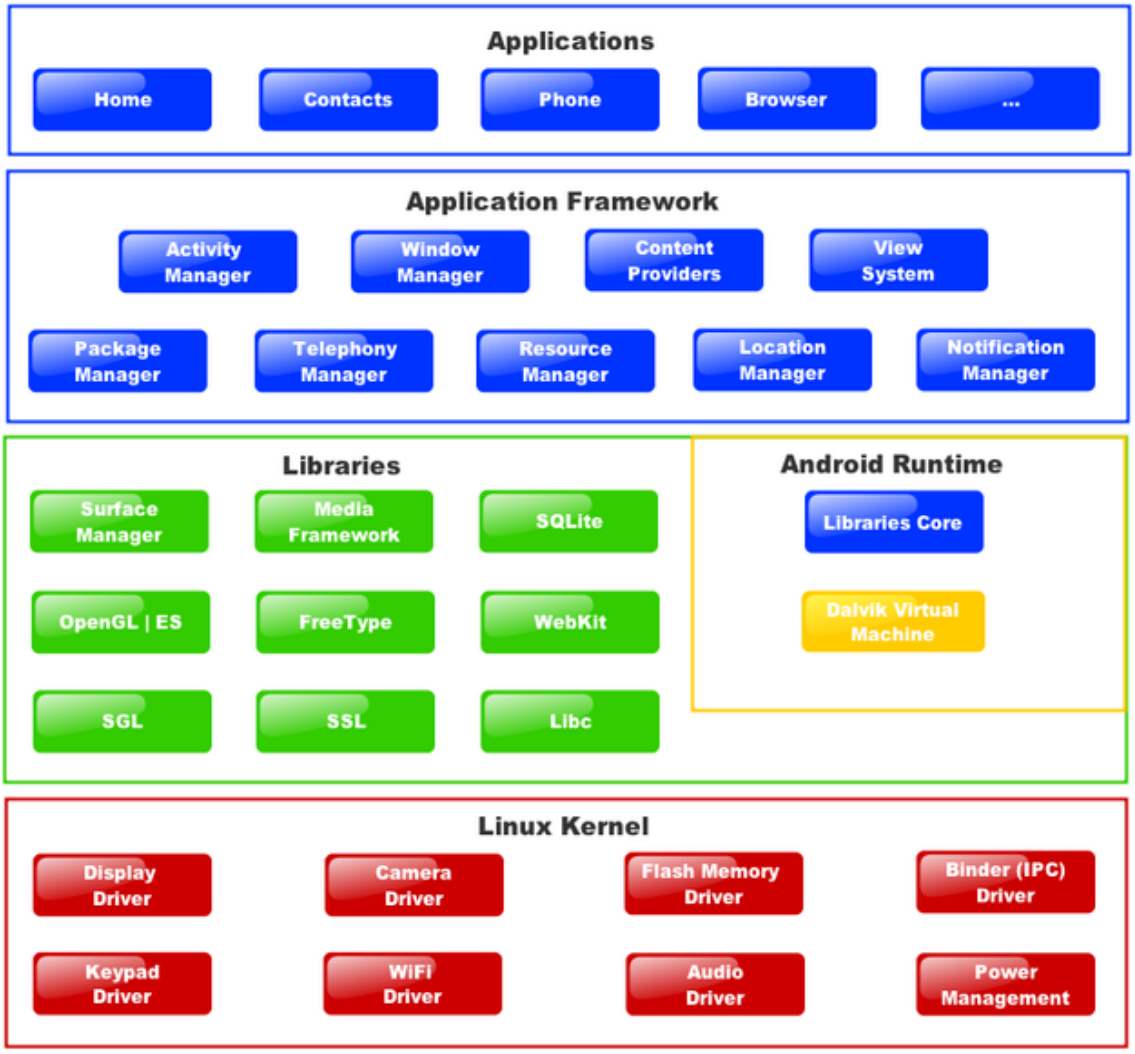
Obrázek 5.7: Zobrazení výsledků dekódované zprávy od klienta

Demonstrační aplikace vyvinutá v rámci této práce je implementována jako kompletní softwarové řešení na platformě *Android* a je tedy zamýšleno ji využívat v mobilních zařízeních. V praxi se tento návrh samozřejmě nehodí pro všechny oblasti užití. V některých případech by byl mnohem vhodnější terminál vybavený přímo čtečkou QR čárových kódů nebo také aplikace pro osobní počítač s webovou kamerou. Systém je však navrhnut tak, aby ho bylo možné provozovat na libovolné platformě. Jedinou podmínkou je dodržení formátu zprávy a všech detailů definovaných standardem (viz kapitola 4.3).

5.4 Použité nástroje

Platforma *Android*

Android je na Linuxu založená softwarová platforma určená hlavně pro mobilní zařízení (chytré telefony, PDA, tablety) [15]. Je vyvíjen společností Google od roku 2005, původně však byl vyvinut společností Android Inc. Vývoj aplikací pro platformu probíhá pomocí Android SDK, což umožňuje vývojářům psát aplikace v jazyce *Java* s využitím knihoven vyvinutých společností Google. Android SDK poskytuje nástroje a API nutné k vývoji aplikací. Platforma zajišťuje především následující funkce:



Obrázek 5.8: Diagram architektury Android

- Aplikační prostředí umožňující opakované použití a nahrazování komponent.

- Virtuální stroj pro běh *Java* aplikací – Dalvik. Dalvik, který má odlišnou architekturu od standardního virtuálního stroje pro Javu (viz obrázek 5.8, staví na podmnožině knihoven projektu Apache Harmony, což je open source projekt, který je přepisem technologií *Java* jako open source. Důsledkem takového řešení je zvýšená nezávislost na původních licencích, které si s sebou původní jazyk i prostředí *Java* nese.
- Integrovaný webový prohlížeč založený na open source jádře *WebKit*.
- Optimalizované grafické funkce pro 2D a 3D grafiku založené na OpenGL ES 1.0 specifikaci (volitelně s hardwarovou akcelerací).
- Podpora SQLite pro efektivní ukládaní strukturovaných dat.
- Podpora pro přehrávání standardních typů medií (obrázků, písniček, videí) v různých formátech (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
- Telefonní funkce pomocí GSM (Globální Systém pro Mobilní komunikaci).
- Bluetooth, EDGE, 3G, a WiFi (záleží na konkretním hardware).
- Volitelně integrovaný fotoaparát, kompas, akcelerometr.
- Bohaté vývojové prostředí včetně emulátoru zařízení a nástrojů pro ladění programu, profilování využití paměti a výkonu.

Knihovna *ZXing*

ZXing (zvaný „zebra crossing“) je open–source multiplatformní knihovna pro zpracovaní 1D/2D čárových kódů implementovaná v Javě. Zaměřuje se na používání vestavěného fotoaparátu v mobilním zařízení a rozpoznání čárového kódu v zařízení, které nemusí komunikovat se serverem. Momentálně podporuje tyto typy čárových kódů:

- UPC-A a UPC-E
- EAN-8 a EAN-13
- Code 39
- Code 93
- Code 128
- QR Code
- ITF
- Codabar
- RSS-14 (všechny varianty)
- Data Matrix
- PDF 417

Kapitola 6

Aplikace systému

V této kapitole budou rozepsány konkrétní oblasti, ve kterých by bylo možné systém popisovaný v této práci v praxi využít. Příkladů by se ovšem dalo najít mnohem více.

6.1 Ověření totožnosti v instituci

Systém přenosu informace vizuální cestou pomocí čárového kódu je možné použít například i v případě ověření totožnosti při návštěvě určité instituce. Pokud navštívíme pobočku bankovní instituce, je potřeba prokázat svoji totožnost – to je v dnešní době řešeno pomocí průkazu totožnosti, který se předloží příslušnému zaměstnanci. V takovém případě hodně záleží na tom, jak svědomitý je personál, který na přepážce pracuje. Cílem by mělo být co nejvíce usnadnit personálu operaci ověření zákazníka. Krom přímého ověření totožnosti se ještě pro autorizaci operací přímo na přepážce používá ověření na základě podpisového vzoru, což ovšem není možné provést okamžitě na místě operace, ale je potřeba znalce písma pro adekvátní ověření. Tyto nedostatky by se daly řešit použitím systému popisovaného v této práci.

Při použití tohoto systému by bylo možné jednoduše ověřit nárok zákazníka provádět požadované operace. Jediným slabým článkem může být případná ztráta či odcizení přístroje s klientskou aplikací systému, ovšem i tak je tato aplikace stále chráněna uživatelským heslem.

Ověření při použití tohoto systému by pak probíhala tak, že by se zákazník za použití speciálního profilu ve své klientské aplikaci autorizoval načtením vygenerovaného QR čárového kódu do čtecího zařízení u přepážky instituce. Čtecí zařízení (tedy serverová aplikace) by pak po ověření přes systém instituce signalizovala, zda má uživatel oprávnění či nikoliv.

Vhodné vlastnosti serverové aplikace

Serverová aplikace k tomuto účelu autorizace – tedy ověření na přepážce – nevyžaduje žádné speciální požadavky. Přesto budou dále uvedeny vhodné parametry pro realizaci.

- Nejsnazším řešením je použití jednoduché webové kamery či hardwarové čtečky čárových kódů pro skenování QR čárového kódu. Tento hardware se musí připojit ke klasickému počítači – personál na pobočce je ve většině případů již vybaven potřebným zařízením.

- Serverová aplikace je poté realizována jako počítačový software, který je přímo napojen na systém dané instituce, a je tedy přímo integrovaný. Díky tomuto řešení je použití systému transparentní.
- Pro případ užití zprávy verze *B* – tedy zabezpečení zprávy pomocí *PIN* kódu – je vhodné realizaci serverové aplikace rozšířit. Rozšíření spočívá v připojení jednoduchého zobrazovacího zařízení (nejčastěji displeje), na kterém bude zobrazován kód pro klienta.
- Případné modifikace řešení jsou závislé na konkrétních požadavcích každé instituce.

Vhodná přenášená data

- Pro kódování zprávy je možné použít libovolnou verzi zprávy. Z hlediska bezpečnosti je však vhodné použít verzi *B*, tedy zabezpečení pomocí *PIN* kódu. Verze *B* je vhodnější z toho důvodu, že zadávání *PIN* kódu nezabere skoro žádný čas a navíc návštěvy institucí neprobíhají tak často. Na druhou stranu tím získáme jistotu, že bezpečnost je opravdu maximální.
- Ve zprávě stačí téměř pro všechny případy přenést pouze identifikátor uživatele. Není tedy potřeba přenášet žádná volitelná data navíc.

6.2 Elektronický zámek

Další oblastí, kde je možné tento systém využít, je vstup do dveří chráněných elektronickým zámkem. K tomuto účelu může sloužit celá řada již existujících řešení. Výhodou tohoto systému je i fakt, že každému uživateli stačí jen přidat další profil v klientské aplikaci, tedy pouze datový soubor. Odpadá tedy nutnost nosit při sobě další předmět (což zabezpečuje vyšší pohodlí).

Vhodné vlastnosti serverové aplikace

Dále bude rozepsán optimální způsob realizace serverové aplikace v souvislosti s otevíráním dveří s elektronickým zámkem. Konkrétně tato oblast využití systému není vhodná pro realizaci, která byla použita v demonstrační serverové aplikaci.

- Navržený systém je nejvhodnější realizovat jako vestavěný systém, tedy systém, který je z vetší části specifické hardwarové řešení se softwarovým dekódováním zprávy.
- Jako jednotku pro skenování QR čárového kódu je vhodné použít hardwarovou čtečku, která na výstup přímo dekóduje čárový kód do textové podoby.
- Softwarové dekódování zprávy může být realizováno pomocí softwarové knihovny podobné knihovně *Cryptor* realizované v rámci této práce. Vestavěný systém je však vhodné realizovat raději v programovém jazyce *C/Asembler*.
- Na základě přenesené zprávy je nutné vyhodnotit právo na otevření elektronického zámku. Tato operace může být realizována několika způsoby:

– První z možností je otevřít dveře na základě obdržení jedinečného kódu dveří. Nevýhodou tohoto způsobu však je, že není možné jednoduše zrušit oprávnění k otevření zámku jednomu uživateli.
– Lepším řešením – ovšem za cenu nutnosti připojeni serverové aplikace alespoň do vnitřní sítě – je ověřovat oprávnění na základě identifikátoru uživatele. Serverová aplikace musí mít přístup k databázi uživatelů s jejich oprávněními. V tomto případě je možné měnit přístupová práva jednotlivým uživatelům mnohem jednoduššeji.

Vhodná přenášená data

- Pro kódování zprávy přenášené při použití jako elektronický zámek je vhodné zvolit zprávu verze *A*, tedy zprávu se zabezpečením relace pomocí časového razítka. Důvodem je především umožnění uživatelům rychlý přístup bez nutnosti zadávat další kódy. Samozřejmě použití verze *A* není podmínkou, pro vyšší bezpečnost je možné použít i zprávy verze *B*.
- Ve většině případů není nutné zprávou přenášet žádná speciální data, postačí pouze nezbytné součásti zprávy (viz kapitola 4.3). Položka *Data* může zůstat nevyplněna.

6.3 Bezdrátová autorizace k terminálu

Velkou výhodou tohoto systému je, že je bezdrátový – přenos informace probíhá konkrétně vizuální cestou. Autorizaci či komunikaci lze obecně provádět například přes výkladní skříň obchodu. Tato vlastnost umožňuje v kombinaci z jinými systémy přihlášení pro účely personalizace k různým terminálům.

Konkrétní specifikace podobného způsobu využití popisovaného systému se odvíjí od daných požadavků, není možné specifikaci jednoduše zobecnit a přesně určit, které parametry jsou nejvhodnější (jak bylo specifikováno u jiných způsobů užití).

6.4 Autorizace u osobního počítače

Systém z této práce lze také využít jako mechanismus přihlášení k účtu u běžného počítače. Může jít o jednoduchý způsob personalizace stolních počítačových stanic v internetových kavárnách nebo třeba ve školách. Celý uživatelský účet je bezpečně uložen na internetovém uložišti a v momentě přihlášení je zaveden do konkrétní stanice. Stejně tak může systém sloužit k přihlášení k domácí stanici v případě, že uživatel vyžaduje větší bezpečnost.

Vhodné vlastnosti serverové aplikace

- Pro skenování QR čárového kódu je nejvhodnější využít webové kamery. Hlavním důvodem je možnost všestranného využití takového hardware a dále nízká cena.
- Zbytek serverové aplikace realizovaný jako počítačový program.

Vhodná přenášená data

- Verze zprávy vhodná pro tento způsob užití je verze A, tedy zpráva zabezpečená pomocí časového razítka. V tomto případě přihlášení je verze A vysoce bezpečná, protože je předpokládáno, že se uživatel u stanice zdrží mnonásobně delší dobu než je doba, za kterou vyprší platnost zprávy pro přihlášení.
- Pro zajištění předchozího bodu je nutné omezit počet přihlášení na jeden účet a pouze na jedno přihlášení.
- Jak již tomu bylo u některých předchozích příkladů využití tohoto systému, není nutné přenášet ve zprávě žadná dodatečná data, postačí pouze identifikátor uživatele.

6.5 Nevhodné oblasti aplikace systému

Jistě existuje mnoho dalších oblastí, kde by se dal systém autorizace na základě bezpečného přenosu informace pomocí čárového kódu využít. Při integraci systému je však nutné brát v potaz některé důležité aspekty. Některé stávající systémy je třeba upravit – přizpůsobit je pro integraci tohoto systému.

Vzhledem k tomu, že systém slouží k zabezpečení dat a využivá na klientské úrovni plně softwarové řešení potýka se systém s problémem který musí řešit všechny softwarové aplikace a to je nelegální kopírování. Systém zábrání kopírování utočníkem, avšak nedokáže předejít případům kdy danou kopii vytvoří uživatel sám. Při řešení tohoto problému záleží především na politice systému. Nejjednoduším řešením je koncipovat využívání systému tak aby nebylo v zájmu uživatele aby prováděl nelegální kopírování nebo aby toto kopírování dat nebylo pro systém problém. Doposud popisované oblasti kde je možné vhodně integrovat tento systém spadali právě do těchto dvou kategorií. Avšak i systémy, které tyto vlastnosti nemají se ve většině případů dají správně modifikovat.

Jízdné v hromadné dopravě

Typickým příkladem kdy velmi záleží na politice systému v závislosti na kopírování je jízdné v hromadné dopravě. V případě městké hromadné dopravy většinou každé větší město využivá vlastního řešení a některá z těchto řešení jsou a jiná nejsou vhodná pro integraci tohoto systému. Aby bylo možné systém popisovaný v této práci integrovat jako způsob účtování za služby dopravních společností je nutné aby politika účtování za služby měla následující vlastnosti.

- Účtování za každou jízdu nebo účtování za jednotku vzdálenosti. V těchto případech je na daný uživatelský účet připisována každá jízda daným dopravním prostředkem. Z toho vyplývá, že i v případě zkopírování, tedy v případě, že jeden učet využívá více uživatelů, je správně započtena každá jízda.

Vhodné vlastnosti serverové aplikace

Při realizaci serverové aplikace sloužící pro použití jako účtovací zařízení při výběru jízdného je vhodné brát v potaz následující specifikaci. Serverová aplikace může mít v tomto případě dvojí podobu.

1. Může být realizována jako aplikace pracovníka vybírající jízdné – například průvodčí ve vlaku.

    - Aplikace realizovaná jako software pro mobilní zařízení.
    - Pro skenování QR kódu využita vestavěná kamera mobilního zařízení.

2. Nebo může být realizována jako terminál – například při průchodu turniketem u vstupu do metra.

    - Aplikace realizovaná jako vestavěný systém. Podobně jak v případě serverové aplikace pro elektronický zámek (viz kapitola 6.2).

Pro obě verze realizace serverové aplikace však platí, že každé použití systému je potřeba zaznamenat na účet uživatele. Vzhledem k tomu, že uživatel může systém využívat vícekrát během dne a na různých místech, je potřeba synchronizovat stav účtu.

Vhodná přenášená data

- Pro přenos dat je vhodné využít kódování zprávy ve verzi A, tedy zprávy zabezpečené pomocí časového razítka. Při použití této verze je lépe zabezpečeno, že autorem zprávy je opravdu daný uživatel.
- V případě učtovaní za jednotlivou jízdu postačí ve zprávě přenést pouze identifikátor uživatele. Pokud je však učtování prováděno na základě třeba jednotky míry, je potřeba dále hodnoty jednotky míry přenášet.

Kapitola 7

Závěr

V této práci jsem se zabýval návrhem systému pro přenos dat vizuální cestou s využitím čárových kódů. Hlavní důraz byl kladen na bezpečnost. Byly navrhnuty různé možnosti zabezpečení dat, a to nejen během jejich přenosu, ale i při jejich uskladnění.

V rámci práce jsem zpracoval přehled různých typů čárových kódů, při jejichž popisu jsem se zaměřil na vlastnosti důležité pro aplikaci v navrhovaném systému. Dále jsem analyzoval různé jiné způsoby přenosu dat a porovnal právě s vizuálním přenosem dat, který byl v práci použit.

Po návrhu jsem implementoval jádro systému, které slouží k realizaci samotného zabezpečeného přenosu dat. Vstupní data transformuje do podoby vhodné pro přenos pomocí čárového kódu, ale také zpět do jejich původní podoby. Při transformaci jsou přenášená data rozšířena o další informace zajišťující požadovanou bezpečnost.

Kromě jádra aplikace vznikly i demonstrační aplikace pro testování možností systému a jejich prezentaci. Pomocí dvou demonstračních aplikací – klientské a serverové – je možné prezentovat přenos informací včetně různých možností zabezpečení.

Ke konci práce jsem navrhl oblasti, kde by bylo možno systém prakticky využít, a to například v aplikaci pro automatické otevírání dveří či při přihlašování k osobnímu počítači. U každého návrhu využití systému je rozvedeno, jak konkrétně integrovat systém, aby splňoval dané požadavky. Systém je od začátku pojat jako koncept, který zajišťuje jádro zabezpečeného přenosu informace pomocí čárového kódu. Specifikace definovaná jako standard umožňuje specifickou implementaci na různých platformách a přitom zachovává vzájemnou kompatibilitu mezi jednotlivými implementacemi.

Systém je navržen tak, aby jej bylo možné dále rozšiřovat, a tím pokrýt ještě větší spektrum oblastí, ve kterých by jej bylo možno úspěšně integrovat a používat. Návrh a implementace jádra systému v této práci však rozsahem obsahuje vše potřebné pro použití v systémech uvedených v kapitole 6. Nemalým přínosem – co se týče rozšíření – by byla implementace klientské aplikace na více mobilních platformách, což by systém činilo použitelným pro široké spektrum uživatelů.

Literatura

[1] EAN–13 specifikace. http://www.barcodeisland.com/ean13.phtml, [Online, přístupné 15.12.2010].

[2] ISO/IEC 16022:2006. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=44230, [Online, přístupné 14.12.2010].

[3] Kód DataMatrix. http://en.wikipedia.org/wiki/Data_matrix_%28computer%29, [Online, přístupné 27.12.2010].

[4] Kód UPC. http://www.makebarcode.com/specs/upc_a.html, [Online, přístupné 15.12.2010].

[5] MIL-STD-1189 Standard Department of Defense Barcode Symbology. https://assist.daps.dla.mil/quicksearch/basic_profile.cfm?ident_number=36123, [Online, přístupné 12.12.2010].

[6] PDF417 symbology. http://www.easesoft.net/PDF417.html, [Online, přístupné 16.12.2010].

[7] Reed-Solomon Codes. http://www.cs.cmu.edu/afs/cs/project/pscico-guyb/realworld/www/reedsolomon/reed_solomon_codes.html, [Online, přístupné 18.12.2010].

[8] Specifikace kódu 128. http://www.barcodeman.com/info/c128.php, [Online, přístupné 23.01.2011].

[9] Specifikace kódu 39. http://www.barcodeman.com/info/c39_1.php, [Online, přístupné 12.01.2011].

[10] Standardy QR kódu. http://www.denso-wave.com/qrcode/qrstandard-e.html, [Online, přístupné 20.12.2010].

[11] Symbologie kódu 93. http://www.barcodeisland.com/code93.phtml, [Online, přístupné 4.12.2010].

[12] Hanáček, P.: Moderní kryptografie, symetrická a asymetrická kryptografie – Skripta k předmětu Kryptografie. https://www.fit.vutbr.cz/study/courses/KRY/private/kry03.pdf, 2008, str. 64. [Online, přístupné 18.12.2010].

[13] Hanáček, P.; Staudek, J.: *Bezpečnost informačních systémů*. ÚSIS, 2000, ISBN 80-238-5400-3, 127 s.

[14] Lindell, J. K. Y.: *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2007, ISBN 1-58488-551-1, 552 s.

[15] Meier, R.: *Professional Android 2 Application Development*. Wrox, 2010, ISBN 978-0-470-56552-0, 543 s.

[16] Palmer, R. C.: *The Bar Code Book*. Helmers Publishing, 1995, ISBN 0-911261-09-5, 386 s.

[17] Stevenson, R.: Laser Marking Matrix Codes on PCBs. `http://pcdandm.com/pcdmag/mag/0512/0512stevenson.pdf`, prosinec 2005, [Online, přístupné 14.12.2010].

Příloha A

Obsah CD

Obsah přiloženého CD:

- Soubor `dp.pdf` – technická zpráva ve formátu PDF.
- Soubor `README` – návod k instalaci.
- Adresář `paper/` – zdrojové soubory technické zprávy (LaTeX).
- Adresář `Cryptor/` – zdrojové texty knihovny *Cryptor* pro generování/dekódování zprávy.
- Adresář `Cryptor/doc/` – programová dokumentace knihovny *Cryptor* vygenerovaná pomocí aplikace *JavaDoc*. Titulní strana se nachází v souboru index.html.
- Adresář `CryptorTest/` – zdrojové texty pro testovaní knihovny *Cryptor*.
- Adresář `MobileCard/` – zdrojové texty demo aplikace pro plaformu Android.