FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Zabezpečený tisk z mobilního telefonu s OS Android s pomocí Bluetooth
Student:	Martin Balko
Vedoucí:	Ing. Pavel Kubalík, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Bezpečnost a informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	do konce letního semestru 2021/2022

Pokyny pro vypracování

Prozkoumejte existující řešení pro Bluetooth tisk.
Analyzujte Bluetooth a bezpečnost Bluetooth komunikace.
Analyzujte a navrhněte řešení, které zvýší samotné zabezpečení Bluetooth protokolu.
Zaměřte se zejména na řešení, které není závislé na použité verzi protokolu Bluetooth.
Navržené řešení se bude skládat ze: serverové části pro Raspberry Pi a uživatelské části pro Android zařízení.
Bluetooth komunikace bude probíhat mezi serverovou a uživatelskou částí.
Navržené zařízení zrealizujte a řádně otestujte.

Elektronicky schválil/a prof. Ing. Pavel Tvrdík, CSc. dne 25. ledna 2021 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalárska práca

Zabezpečený tisk z mobilního telefonu s OS Android s pomocí Bluetooth

Martin Balko

Katedra počítačových systémů
Vedúci práce: Ing. Pavel Kubalík, Ph.D.

2. mája 2021

Poďakovanie

Ďakujem môjmu vedúcemu práce za praktické rady, prínosné konzultácie a celkovú pomoc v priebehu tvorby práce. Takisto ďakujem svojej rodine a priateľom za podporu.

Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval(a) samostatne a že som uviedol(uviedla) všetky informačné zdroje v súlade s Metodickým pokynom o etickej príprave vysokoškolských záverečných prác.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb, autorského zákona, v znení neskorších predpisov. V súlade s ustanovením § 46 odst. 6 tohoto zákona týmto udeľujem bezvýhradné oprávnenie (licenciu) k používaniu tejto mojej práce, a to vrátane všetkých počítačových programov ktoré sú jej súčasťou alebo prílohou a tiež všetkej ich dokumentácie (ďalej len „Dielo“), a to všetkým osobám, ktoré si prajú Dielo používať. Tieto osoby sú oprávnené Dielo používať akýmkoľvek spôsobom, ktorý neznižuje hodnotu Diela, ale len pre nezárobkové účely. Toto oprávnenie je časovo, územne a množstevne neobmedzené.

V Prahe 2. mája 2021 . . . . . .

Odkaz na túto prácu

Balko, Martin. *Zabezpečený tisk z mobilního telefonu s OS Android s pomocí Bluetooth*. Bakalárska práca. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Abstrakt

Práca sa zameriava na návrh riešenia pre bezpečnú Bluetooth tlač pomocou Android zariadení a Raspberry Pi. Súčasťou práce je analýza existujúcich riešení, analýza technológie Bluetooth a jej bezpečnosti. Pri návrhu riešenia je kladený dôraz na zlepšenie bezpečnosti technológie Bluetooth nezávisle na jej verzii. Navrhnuté riešenie využíva protokol Diffie-Hellman s využitím eliptických kriviek, šifru AES, hash funkciu SHA a digitálne certifikáty. Riešenie je implementované pomocou programovacích jazykov Python a Kotlin. Súčasťou práce je aj testovanie implementovaného riešenia.

Klíčová slova Bluetooth tlač, bezpečná komunikácia, šifrovanie, Android aplikácia, Raspberry Pi, Python, Kotlin

Abstract

This bachelor thesis deals with the design and implementation of secure Bluetooth print using Android device and Raspberry Pi. Part of it consists of the analysis of existing solutions, analysis of Bluetooth, and its security. The design emphasis on such a solution which improves overall Bluetooth security. The designed solution uses Elliptic-curve Diffie-Hellman, cipher AES, hash function SHA, and digital certificates. The solution is programmed using Python and Kotlin. The final part of this thesis consists of testing of the implemented solution.

Keywords Bluetooth printing, secure communication, encryption, Android application, Raspberry Pi, Python, Kotlin

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Bezdrôtová komunikácia hrá v súčasnej dobe dôležitú úlohu vo väčšine aspektov technologického sveta. Pomocou bezdrôtových komunikačných technológií sa človek v súčasnosti pripája k najrôznejším nástrojom dennodenného použitia. Vďaka tomuto sa pre každého užívateľa stávajú jednotlivé produkty a služby prístupnejšie a jednoduchšie na používanie. Mobilné telefóny sa radia medzi hlavné zariadenia, ktoré užívatelia používajú pre bezdrôtovú komunikáciu s rôznymi produktami a službami.

Zatiaľ čo mobilné telefóny dokážu zvládať čoraz viac funkcií, tlač dokumentov sa radí medzi tie oblasti, ktoré bývajú značne ignorované. Vďaka tomuto dôvodu existuje veľmi obmedzené množstvo spôsobov, akým je užívateľ schopný využiť svoj mobilný telefón pre tlač dokumentov. Pre takýto spôsob využitia je veľmi vhodná technológia Bluetooth, ktorá má aj množstvo iných existujúcich využití. Práve pre tieto dôvody som sa rozhodol vybrať si túto tému práce.

V práci sa zaoberám analýzou, návrhom, implementáciou a testovaním aplikácie, ktorá sa vyššie načrtnutý problém snaží vyriešiť. Jedná sa o užívateľskú aplikáciu pre Android zariadenia so serverovou časťou pre Raspberry Pi. Tieto dve súčasti vďaka vzájomnej komunikácií umožňujú užívateľovi tlačiť dokumenty pomocou technológie Bluetooth jednoducho a bezpečným spôsobom priamo z ich mobilného telefónu s operačným systémom Android.

V prvej časti práce sa zaoberám analýzou existujúcich riešení pre tlač dokumentov pomocou technológie Bluetooth. Následne analýzou samotnej technológie Bluetooth a jej bezpečnosti zisťujem jej možné zraniteľnosti. Analýzou možných riešení v ďalšej časti navrhujem také riešenie, ktoré bude jednak zvyšovať bezpečnosť technológie Bluetooth a zárobeň bude vhodné na použitie pre zariadenia s operačným systémom Android a Raspberry Pi. V predposlednej časti práce dokumentujem implementáciu navrhnutého riešenia. Nakoniec túto implementáciu testujem a toto testovanie riadne dokumentujem v záverečnej časti práce.

Kapitola 1

Cieľ práce

Cieľom bakalárskej práce je vytvoriť riešenie pre bezpečnú tlač dokumentov z mobilného zariadenia s operačným systémom Android pomocou komunikačnej technológie Bluetooth.

Cieľom rešeršnej časti práce je preskúmať a analyzovať existujúce riešenia pre tlač dokumentov pomocou technológie Bluetooth a ich kompatibilitu s Android zariadeniami. Medzi ďalšie ciele tejto časti patrí analýza technológie Bluetooth so zameraním na jej bezpečnosť a nájdenie jej možných slabín. Súčasťou je aj navrhnúť riešenie, ktoré bezpečnosť komunikácie cez Bluetooth zvýši. V rámci tohto návrhu bude dôležitý prieskum a analýza možných bezpečnostných variant. Medzi dôležité vlastnosti tohto bezpečnostného riešenia bude nezávislosť voči používanej verzii technológie Bluetooth. Celkové riešenie sa bude skladať zo serverovej časti pre Raspberry Pi a užívateľskej časti pre zariadenia s operačným systémom Android. Serverová časť bežiaca na Raspberry Pi bude s tlačiarňou komunikovať pomocou rozhrania USB.

Cieľom praktickej časti je navrhnúť rozhranie pre užívateľskú časť navrhnutého riešenia, zvoliť vhodnú implementačnú platformu pre obe časti vytvoreného návrhu a formu výmeny dát medzi užívateľskou a serverovou časťou. Dôležitou súčasťou praktickej časti je aj popis a dokumentácia nadviazania spojenia a samotnej komunikácie medzi užívateľskou a serverovou časťou.

Užívateľská časť, aplikácia pre zariadenia s operačným systémom Android, má umožňovať pripojenie k serverovej časti a bezpečné nadviazanie komunikácie. Nasledujúca komunikácia so serverovou časťou bude zabezpečená po celú dobu riešením navrhnutým v rešeršnej časti a implementovaným v praktickej časti práce.

Kapitola 2

Analýza

V rámci tejto časti mojej práce postupne preskúmam existujúce riešenia pre tlač dokumentov pomocou technológie Bluetooth. Následne analyzujem a predstavím operačný systém Android, ktorý bude dôležitý pre užívateľskú časť navrhovanej aplikácie a radu jednodoskových počítačov Raspberry Pi, ktorej niektorý predstaviteľ bude v rámci navrhovanej aplikácie fungovať ako serverová časť. Následnou analýzou technológie Bluetooth a jej samotnej bezpečnosti zistím potencionálne bezpečnostné zraniteľnosti, na ktoré si počas návrhu riešenia budem dávať pozor. Diskusiou rôznych spôsobov zabezpečenia navrhovanej aplikácie túto časť zakončím.

2.1 Existujúce riešenia

V tejto sekcii predstavím existujúce riešenia pre tlač dokumentov pomocou technológie Bluetooth z Android zariadení. Tlač dokumentov je špecifická záležitosť, preto medzi existujúce riešenia možno zaradiť tlačiarne, ktoré majú podporu Bluetooth už od výroby a špecialne navrhnuté adaptéry, ktoré túto podporu pridávajú.

2.1.1 Bluetooth tlačiarne

Medzi existujúce riešenia problému ako tlačiť dokumenty pomocou technológie Bluetooth patria tlačiarne so zabudovanými komponentami umožňujúcimi komunikáciu pomocou technológie Bluetooth. Takéto tlačiarne sú ale veľmi často drahšie ako klasické tlačiarne. Ich dokumentácia taktiež veľmi často nezahrňuje detailné informácie o verzii technológie Bluetooth alebo spôsobe, akým tlačiareň so zariadením komunikuje. Užívateľ v takom prípade nemá istotu, že pri odosielaní dokumentov pre tlač je komunikácia medzi jeho zariadením a tlačiarňou dostatočne zabezpečená.

Medzi ďalšie nevýhody patrí fakt, že nie je zabezpečená plná kompatibilita medzi tlačiarňou a Android zariadením užívateľa.

2.1.2 USB Bluetooth adaptéry

Druhým existujúcim riešením sú externé USB Bluetooth adaptéry. Zapojením takéhoto adaptéru do USB portu na tlačiarni je možné takejto tlačiarni pridať podporu komunikácie cez Bluetooth.

Obr. 2.1: USB Bluetooth adaptér. Prevzaté z [1]

Tieto adaptéry však nie sú primárne koncipované pre zapojenie do tlačiarní, ale do počítačov bez integrovanej podpory technológie Bluetooth. Preto je takéto riešenie často hlavne u starších tlačiarní nekompatibilné a nie je zaručené správne fungovanie. Medzi ďalšie nevýhody patrí fakt, že väčšina tlačiarni používa pre komunikáciu pomocou USB zakončenia typu USB-B, zatiaľ čo drvivá väčšina externých USB Bluetooth adaptérov (vďaka tomu že sú koncipované pre používanie s počítačmi) je vyrábaná s USB zakončením typu USB-A. [1]

Obr. 2.2: Zakončenia USB-B a USB-A. Foto autor

V priebehu mojej práci som nenašiel žiadne existujúce riešenie, ktoré by spĺňalo bezpečnostné požiadavky. Taktiež som nenašiel také existujúce riešenie, ktoré by bolo špecificky určené pre použitie s Android zariadeniami.

2.2 Android

V tejto časti práce predstavím operačný systém Android, pre ktorý v neskorších kapitolách práce navrhujem a implementujem užívateľskú časť riešenia problému načrtnutého v úvode práce. Údaje čerpám z [2], ak nie je uvedené inak.

Pojmom Android sa označuje operačný systém s otvoreným zdrojovým kódom (zdrojový kód je verejne dostupný) založeným na jadre Linux. Podľa údajov dostupných na [3] sa jedná o najpoužívanejší operačný systém pre mobilné zariadenia. Aj vďaka tejto skutočnosti, spoločne s faktom, že sám vlastním niekoľko zariadení s operačným systémom Android, som sa rozhodol pre návrh užívateľskej časti pre tento operačný systém. Z tabuľky nižšie možno vyčítať, že Android nestráca na význame medzi operačnými systémami pre mobilné zariadenia a pevne si drží svoje prvenstvo. Zatiaľ čo Android využíva viac ako 70 percent mobilných zariadení, druhý najviac používaný operačný systém, iOS vyvíjaný spoločnosťou Apple, je využívaný menej ako 30 percentami mobilných zariadení [3].

Rok	Android	iOS	Ostatné
2015	64.2%	20.2%	15.6%
2016	69.11%	19.29%	11.6%
2017	72.63%	19.65%	7.72%
2018	75.45%	20.47%	4.08%
2019	75.47%	22.71%	1.82%
2020	73.06%	26.28%	0.66%

Tabuľka 2.1: Používanie OS (2015–2020). Údaje prevzaté z [6]

Operačný systém Android má širokú škálu využitia. Okrem mobilov a tabletov sa využíva aj na vývoj inteligentných hodiniek, inteligentných televízorov alebo autorádií.

2.2.1 Vývoj aplikácií

Pre vývoj Android aplikácií je možné vybrať si zo širokej škály programovacích jazykov. Medzi oficiálne podporované a najpoužívanejšie sa radia programovacie jazyky *Java* a *Kotlin*. Zatiaľ čo Java existuje ako programovací jazyk dlhšie, Kotlin sa stal populárnejším a odporúča ho ako programovací jazyk pre vývoj Android aplikácií už aj samotný *Google*. [4]

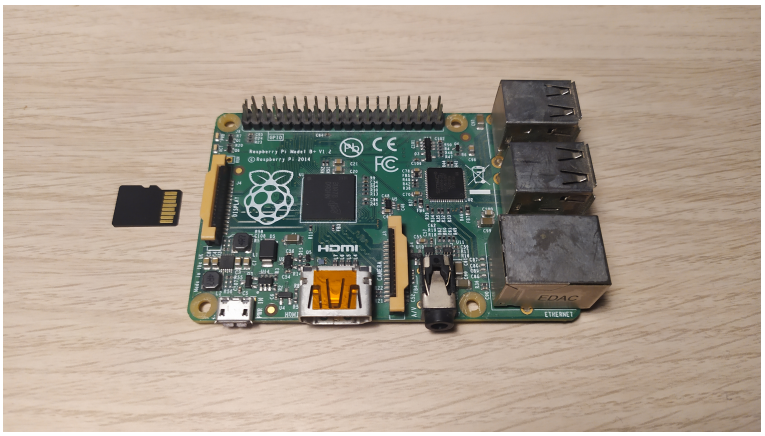
Medzi hlavné výhody programovacieho jazyka Kotlin patrí fakt, že okrem zjednodušenej syntaxe oproti jazyku Java prináša možnosť využívať všetky knižnice a funkcie jazyka Java. Okrem toho je Kotlin celkovo jednoduchší a rýchlejší. [5]

Oba programovacie jazyky patria medzi oficiálne podporované jazyky pre Android. Vďaka tejto skutočnosti existuje množstvo oficiálnych knižníc, pomocou ktorých je možné pracovať s rôznymi funkciami operačného systému Android a samotného zariadenia. Medzi tieto knižnice patria aj také, ktoré pracujú s technológiou Bluetooth a umožňujú programovanie komunikácie cez túto technológiu. Práve tieto knižnice sú dôležitou súčasťou tejto práce. Ďalšími dôležitými knižnicami sú knižnice umožňujúce programovanie bezpečnostných prvkov aplikácie. Tieto a ďalšie sú dôkladnejšie opísané v ďalších kapitolách práce.

2.3 Raspberry Pi

Raspberry Pi (niekedy označované aj skratkou *RPi*) predstavuje radu jednodoskových počítačov. Táto rada je vyvíjaná nadáciou *Raspberry Pi Foundation* z Veľkej Británie. Primárnym účelom týchto počítačov je rozšírenie záujmu o programovanie medzi deťmi v školách a využitie vo výukovom prostredí. Avšak vďaka veľkosti, ponúkanému výkonu, cene a spotrebe sa stali vhodným riešením aj pre zložitejšie problémy. [8]

Všetky počítače rady Raspberry Pi sú uspôsobené na fungovanie s použitím micro SD karty. Z takejto karty je načítavaný nahraný operačný systém. Na kartu sú taktiež ukladané všetky súbory a dáta vytvárané behom používania daného počítača. [9]



Obr. 2.3: Micro SD karta a Raspberry Pi 2. Foto autor

Podľa informácií dostupných v [7, 12] využíva *Raspberry Pi Foundation* pre vývoj ARM procesory. To okrem iného znamená, že pre správne fungovanie ich vydaných počítačov je nutné na micro-SD kartu nahrať operačný systém uspôsobený na fungovanie na ARM procesoroch.

Samotná nadácia okrem vývoja rady Raspberry Pi vyvíja a pravidelne vydáva nové verzie operačného systému *Raspberry Pi OS* (pôvodne nazývaného

Raspbian), ktorý je vyvíjaný a určený na použitie práve s počítačmi rady Raspberry Pi. Pre správne fungovanie RPi počítačov nie je ale použitie *Raspberry Pi OS* nutné. Existuje mnoho ďalších operačných systémov, ktoré spoločne s RPi počítačmi fungujú. [10]

2.3.1 Podpora Bluetooth

Pre moju prácu je dôležitá podpora Bluetooth na užívateľskej aj serverovej časti návrhu. Preto sa v tejto časti zameriavam a analyzujem podporu technológie Bluetooth v jednotlivých modeloch Raspberry Pi. Údaje čerpám z [6], pokiaľ nie je uvedené inak.

Prvým počítačom vydaným v roku 2012 bol *Raspberry Pi 1 Model B*. Tento model podporu Bluetooth komunikácie neobsahoval, rovnako ako niekoľko ďalších vydaných modelov. Od tej doby bolo avšak vyvinutých a vydaných ďalších 10 modelov. V tabuľke nižšie možno vidieť súhrnný zoznam modelov rady Raspberry Pi spoločne s rokom vydania a ich podporu technológie Bluetooth.

Model	Rok vydania	Bluetooth
Raspberry Pi 1 B	2012	Nie
Raspberry Pi 1 B+	2014	Nie
Raspberry Pi 1 A+	2014	Nie
Raspberry Pi 2	2015	Nie
Raspberry Pi Zero	2015	Nie
Raspberry Pi 3	2016	Áno
Raspberry Pi Zero W	2017	Áno
Raspberry Pi 3 B+	2018	Áno
Raspberry Pi 3 A+	2018	Áno
Raspberry Pi 4	2019	Áno

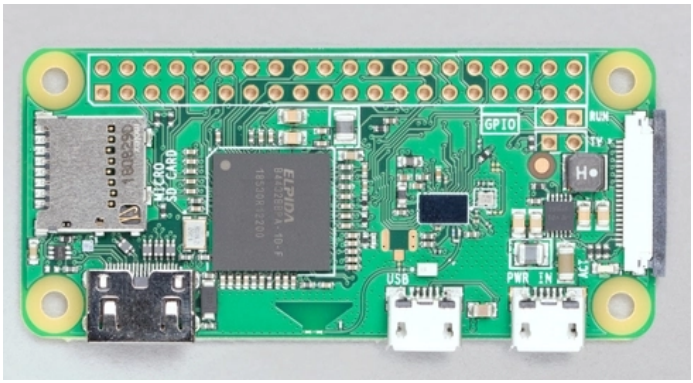
Tabuľka 2.2: Zoznam modelov RPi a ich podpora BT. Údaje prevzaté z [6]

Z tabuľky vyššie možno vyčítať, že od roku 2016, v ktorom bolo vydané *Raspberry Pi 3*, podporujú technológiu Bluetooth všetky novšie modely.

2.3.2 Výber modelu

V rámci mojej práce využívam Raspberry Pi ako serverovú časť, ktorá pomocou technológie Bluetooth komunikuje s Android zariadením. Zároveň komunikuje s tlačiarňou, ktorá je pripojená pomocou USB. Samotný užívateľ fyzicky so serverovou časťou nebude narábať – počítač bude výhradne pripojený len k tlačiarni pomocou USB, nebude k nej pripojená klávesnica, myš ani monitor. Medzi faktory pre výber vhodného modelu Raspberry Pi preto zahŕňam okrem podpory Bluetooth aj veľkosť samotného modelu.

Z údajov diskutovaných vyššie je zrejmé, že v rámci podpory Bluetooth možno vyberať z modelov od roku 2016. Medzi tieto modely sa radí aj *Raspberry Pi Zero W* z roku 2017. Zaujímavou vlastnosťou tohto modelu je, že sa jedná o rozšírenie *Raspberry Pi Zero* vydaného v roku 2015, ktoré bolo a v súčasnosti aj je (spoločne s *RPi Zero W*) najmenším modelom rady Raspberry Pi. [11]



Obr. 2.4: Raspberry Pi Zero W. Prevzaté z [11]

Raspberry Pi Zero W prichádza s podporou pre *Bluetooth 4.1*. Táto informácia je dôležitá pre analýzu bezpečnosti Bluetooth komunikácie v neskorších častiach práce. Okrem toho podporuje aj bezdrôtové pripojenie k internetu a pripojenie externých zariadení (v mojom prípade tlačiarne) pomocou micro USB. [11]

2.4 Bluetooth

Nasledujúca časť obsahuje predstavenie a analýzu technológie Bluetooth a jej bezpečnosti. Údaje čerpám z [13], ak nie je uvedené inak.

Bluetooth (označovaný aj ako *BT*) je technológia pôvodne vyvinutá spoločnosťou *Ericsson* v roku 1994. Bola štandardizovaná pod *IEEE 802.15* v roku 1999. Momentálne je ešte stále aktívne vyvíjaná, avšak o jej vývoj sa v súčasnosti stará organizácia *Bluetooth Special Interest Group* (skrátene *SIG*).

Jedná sa o bezdrôtovú technológiu určenú pre prenos dát medzi dvoma a viac zariadeniami na krátku vzdialenosť. K tomuto účelu je využívaná rádiová komunikácia, presnejšie frekvenčné pásmo 2.4000 GHz až 2.4835 GHz. Toto pásmo Bluetooth zdieľa s ďalšími technológiami. Postupom času sa BT osvedčil a v súčasnosti je využívaný v širokej rade zariadení, medzi ktoré okrem mobilov a tabletov patria aj inteligentné hodinky, automobily, tlačiarne, klávesnice, počítačové myši, slúchadlá a podobne.

Základný koncept komunikácie cez BT tvorí vytvorenie väzby medzi zariadeniami. Tento proces je tiež označovaný ako Bluetooth párovanie. Zariadenia,

ktoré nie sú spárované, nedokážu medzi sebou komunikovať.

Rovnako, ako je každé zariadenie s prístupom na internet identifikované už od výroby svojou MAC adresou, aj zariadenia podporujúce BT majú svoj unikatný identifikátor. Tento identifikátor sa nazýva *Bluetooth adresa* (niekedy tiež označovaná ako *BD_ADDR*). Podobne ako u MAC adresy, jedná sa o 48 bitov informácie, obvykle reprezentovaných ako 12 miestne hexadecimálne číslo. [15]

V súčasnosti existujú dva základné typy technológie Bluetooth – *Bluetooth Classic* (tiež označovaný ako Bluteooth BR/EDR, Bluetooth High Rate, a podobne) a *Bluetooth Low Energy* (často označovaný ako *BLE*, pôvodný názov *Bluetooth Smart*). Zásadný rozdiel medzi týmito typmi je ich určenie. Bluetooth Classic je pokračovanie pôvodnej technológie Bluetooth, BLE je zamerané na použitie so zariadeniami, medzi ktorými dochádza ku krátkej výmene malého množstva dát. BLE býva využívané napríklad pri spojení s inteligentnými hodinkami alebo zariadeniami v domácnosti (teplomery, svetlá a podobne).

Bluetooth Low Energy bolo predstavené v roku 2011. Medzi jeho hlavné výhody patrí malá spotreba energie, čo sa odzrkadľuje pri životnosti zariadení, ktoré čerpajú energiu z batérií. To dosahuje BLE pomocou spôsobu, ako zariadenia medzi sebou komunikujú. Samotné spojenie totiž trvá len po dobu prenosu dát. Tie sú samé o sebe (pri správnom používaní) malej veľkosti, vďaka čomu tieto spojenia trvajú len jednotky milisekúnd. Zariadenia (napríklad domáce teplomery alebo inteligentné hodinky) vďaka nízkej spotrebe BLE dokážu fungovať oveľa dlhšie ako pri používaní BT Classic, pri ktorom spojenia bývajú nepretržité. Avšak medzi jeho nevýhody patrí, že je určený pre mešie objemy dát a aj napriek desiatim rokom jeho existencie je stále zložitejší na implementáciu ako vyše dvadsaťročný BT Classic. [16]

Bluetoth Classic je využívaný medzi zariadeniami a procesmi, ktoré vyžadujú neustále spojenie, prípadne medzi ktorými dochádza k prenosu väčšieho množstva dát. Medzi takéto zariadenia patria reproduktory, klávesnice, tlačiarne a podobne. Medzi nevýhody patrí s ohľadom na BLE väčšia spotreba energie, čo je zas vykúpené vyššou prenosovou rýchlosťou a jednoduchšími, rokmi overenými spôsobmi implementácie.

Z vyššie uvedeného je zrejmé, že pre návrh riešenia v mojej práci je ideálnejší Bluetooth Classic. Odhliadnuc od faktu, že už existujúce riešenia (BT tlačiarne) využívajú BT Classic, pre návrh je dôležitá rýchlosť prenosu a fungovanie implementácie. BT Classic je vďaka svojmu veku stále rozšírenejší a podporovaný väčším množstvom zariadení ako BLE.

2.4.1 Bezpečnosť Bluetooth Classic

Zásadným krokom pri nadväzovaní komunikácie pomocou BT je samotný akt párovania zariadení. Behom párovania je na oboch zariadeniach vytváraný spoločný šifrovací kľúč (nazývaný *Link Key*), ktorý je využívaný pri následnej

autentifikácii a komunikácii medzi zariadeniami. Vďaka tomu vzniká medzi zariadeniami behom párovania forma dôvery.

Behom vývoja BT sa postupne vyvíjali aj spôsoby, kedy sa objavujú bezpečnostné procedúry behom nadväzovania komunikácie medzi zariadeniami. Vďaka tomu existujú celkovo 4 *bezpečnostné módy*, pričom každé zariadenie musí pracovať v jednom z týchto módov. Každý z týchto módov určuje, v akom momente sú bezpečnostné procedúry prvýkrát aplikované. V tabuľke nižšie možno vidieť jednotlivé bezpečnostné módy a úrovne, v ktorých sú bezpečnostné procedúry aplikované.

Bezpečnostný mód	Úroveň
4	Linková
3	Služby
2	Linková
1	Žiadna

Tabuľka 2.3: Prehľad BT bezpečnostných módov. Údaje prevzaté z [13]

Bezpečnostné procedúry sa môžu objaviť celkovo na dvoch úrovňach. Prvá z nich, linková úroveň, označuje moment vytvárania fyzického spojenia medzi zariadeniami. Druhá, úroveň služby, označuje stav, kedy je fyzické spojenie už nadviazané, ale logické kanály ešte nie sú plne vytvorené.

Bezpečnostný mód 1 ako jediný neaplikuje bezpečnosť na žiadnej úrovni. To znamená, že necháva komunikáciu otvorenú a nechránenu voči útokom. Rovnako umožňuje pripojenie akémukoľvek zariadeniu. Všetky verzie BT tento mód podporujú, avšak len z dôvodu spätnej kompatibility so staršími zariadeniami.

Bezpečnostné módy 1 až 3 môžu byť aktívne využívané zariadeniami s BT verziou 2.0 a nižšou. Zariadenia s BT verziou 2.1 a vyššou však aktívne využívajú výhradne bezpečnostný mód 4, ostatné 3 módy podporujú len z dôvodu spätnej kompatibility so staršími verziami BT.

Dôvodom pre takéto chovanie sú 2 rozdielne metódy párovania BT zariadení. Obe metódy majú za úlohu generovanie spoločných šifrovacích kľúčov, avšak každá z nich využíva iné spôsoby, šifry a interakcie s užívateľom. Bezpečnostné módy 2 a 3 využívajú pre párovanie metódu *PIN Pairing* (tiež nazývanú ako *Legacy Pairing*), zatiaľ čo bezpečnostný mód využíva metódu *Secure Simple Pairing* (skrátene *SSP*), ktorá bola vydaná spoločne s bezpečnostným módom 4 vo verzii 2.1.

PIN/Legacy Pairing využíva pre výpočet spoločného šifrovacieho kľúča na párovaných zariadeniach tajný PIN. Tento PIN zadáva užívateľ na všetky párované zariadenia, musí byť identický a mal by ostať utajený. V prípade, že jeho hodnota nedosahuje veľkosť 16 bajtov, je dopĺňaná Bluetooth adresou zariadenia. PIN môže byť reprezentovaný ako 1 až 16 bajtov, avšak obvykle je užívateľom zadávaný ako kombinácia čísel, najčastejšie štvorciferná.

Secure Simple Pairing vydaný v BT verzii 2.1 a vylepšený vo verzii 4.1 v porovnaní s PIN/Legacy Pairing zjednodušuje proces párovania a zvyšuje jeho bezpečnosť použitím eliptických kriviek. Súčasťou SSP sú 4 *asociačné modely*, ktoré sú použité na základe schopnosti zariadenia prijímať a zobrazovať informácie.

Numeric Comparison je asociačným modelom využívaným v situáciach, kedy sú obe párované zariadenia schopné zobrazovať aj prijímať informácie. Behom párovania užívateľ na oboch zariadeniach potvrdí, prípadne vyvráti rovnosť zobrazovanej informácie, šesťciferného čísla, čím umožní pokračovanie párovania, respektíve jeho zlyhanie. Zobrazované šesťciferné číslo nie je použité pri výpočte šifrovacieho kľúča, čím nevzniká možnosť pre prípadného útočníka zneužiť túto informáciu.

Passkey Entry je asociačný model navrhnutý pre situácie, kedy má iba jedno zariadenie schopnosť prijímať informácie zadávané užívateľom, zatiaľ čo druhé je schopné informácie zobrazovať. Behom párovania zariadenie schopné zobrazovať informácie opäť zobrazí šesťciferné číslo. Užívateľ následne túto informáciu zadá do zariadenia, ktoré je schopné zadávané informácie prijať. V prípade, že užívateľ zadá číslo nesprávne, párovanie zlyhá.

Just Works je asociačný model využívaný v situáciach, kedy aspoň jedno zariadenie nie je schopné informácie zobrazovať ani prijímať. Behom párovania prebieha rovnaký proces ako pri asociačnom modeli *Numeric Comparison*, avšak kvôli absencii displeja na jednom zariadení nie je schopný overiť rovnosť používaných čísel. Vďaka tomu model neposkytuje ochranu proti Man-In-The-Middle (MITM) útokom.

Out Of Band je posledným asociačným modelom SSP. Býva využívaný v spolupráci s ďalšími technológiami, ktoré sú využívané pri párovaní. Takýmito technológiami môže byť napríklad NFC a použitie NFC karty. Pri používaní tohto asociačného modelu je dôležité overiť a zabezpečiť ochranu voči odposluchu a MITM útokom na všetkých ďalších technológiach, ktoré sa pri párovaní využívajú.

2.4.2 Zabezpečenie na aplikačnej vrstve

Napriek rozsiahlej škále opatrení, ktoré organizácia Bluetooth SIG pre technológiu Bluetooth vyvinula, neustále existujú spôsoby a potenciálne hrozby, ktoré bezpečnosť komunikácie ohrozujú. Kvôli tomu je opodstatnené pri implementácií komunikácie pomocou BT pridať ďalšie bezpečnostné opatrenia nezávislé na zariadení a dostupnej verzií BT. Ako správna vrstva sa zdá byť

vrstva aplikačná. Pridaním ďalšej bezpečnosti na túto vrstvu dosiahneme nezávislosť voči bezpečnostným konceptom BT, keďže dáta posúvané tejto technológii už budú ošetrené a ochránené proti možným slabinám, ktoré Bluetooth obsahuje. [14]

2.5 Bezpečná komunikácia

V predchádzajúcich častiach práce som uviedol, že ako serverovú časť využívam *RPi Zero W*, ktoré podporuje *Bluetooth 4.1*. To znamená, že pri BT párovaní serverovej časti s iným zariadením je použitý mechanizmus Secure Simple Pairing. K serverovej časti návrhu nie je pripojená klávesnica ani myš. Užívateľ fyzicky nemanipuluje so serverovou časťou, čo znamená, že je využívaný asociačný model *Just Works*, ktorý neposkytuje ochranu proti MITM útokom. Vďaka tomu je v rámci návrhu riešenia v tejto práci dôležité navrhnúť také zabezpečenie komunikácie na aplikačnej vrstve, ktoré eliminuje možnosť MITM útoku. Navrhované riešenie bude rovnako potrebné navrhnúť tak, aby spoľahlivo fungovalo aj s použitím iných verzií BT. To sa docieli zabezpečením na aplikačnej vrstve z dôvodov predstavených v predchádzajúcej časti. V tejto časti práce diskutujem rôzne spôsoby zabezpečenia komunikácie, ich výhody a nevýhody.

2.5.1 Absencia šifrovania

Ako prvý spôsob zabezpečenia komunikácie cez BT sa ponúka úplná absencia šifrovania na aplikačnej vrstve. Znamenalo by to, že celkové zabezpečenie komunikácie by spadalo pod BT. Vďaka tomu by sa užívateľ spoliehal na to, že bezpečnosť je dostačujúca a v súčasnosti nehrozí žiadna možnosť úniku dát, prípadne celkového prelomenia zabezpečenia BT. Na to sa ale spoliehať nedá. V mojom prípade, za použitia *RPi Zero W* ako serverovej časti navrhovaného riešenia, Bluetooth nezaisťuje ochranu proti MITM útokom. Tým pádom tento spôsob nemožno využiť.

2.5.2 Šifrovanie komunikácie

Údaje v tejto časti čerpám z [17], ak nie je uvedené inak.

Ďalšou možnosťou zabezpečenia komunikácie je šifrovanie dát, ktoré si zariadenia medzi sebou pomocou BT vymieňajú. Pomocou šifrovania na aplikačnej vrstve navrhovaného riešenia budú dáta odovzdávané technológii Bluetooth už zašifrované. Konečné dešifrovanie znova prebieha na aplikačnej vrstve po prijatí dát. Týmto spôsobom sú dáta chránené nezávisle na bezpečnostných opatreniach technológie Bluetooth. Šifrovanie komunikácie sa javí ako správne bezpečnostné opatrenie, ktoré môže byť do návrhu zapojené, avšak prináša niekoľko problémov, ktoré je nutné adresovať.

Prvým z problémov šifrovania komunikácie je výber správneho typu šifrovacieho algoritmu, ktoré bude riešenie využívať. Šifrovacie algoritmy sa delia na dva základné typy: *symetrické* a *asymetrické*. Hlavným rozdielom medzi týmito typmi šifrovacích algoritmov je spôsob šifrovania a dešifrovania dát.

Pri použití symetrického šifrovania je pre proces šifrovania aj dešifrovania dát využívaný rovnaký *šifrovací kľúč*. Takéto šifrovacie algoritmy bývajú v porovnaní s *asymetrickým* šifrovaním rýchlejšie. Odporúčaným symetrickým šifrovacím algoritmom je podľa [18] *Advanced Encryption Standard* (skrátene označovaný *AES*). Symetrické šifrovanie sa využíva napríklad pre správu a šifrovanie, respektíve dešifrovanie väčšieho množstva dát.

Asymetrické šifrovanie funguje na princípe *verejného* a *súkromného* kľúča. Pre šifrovanie dát sa využíva verejný kľúč. Takto zašifrované dáta je možné dešifrovať súkromným kľúčom, ktorý sa s verejným kľúčom nezhoduje. Vďaka tomu je možné verejný kľúč distribuovať bez akejkoľvek bezpečnostnej hrozby. V porovnaní so symetrickým šifrovaním je tento typ šifrovania pomalší, avšak vďaka možnosti distribúcie verejného kľúča bezpečnejší. Využíva sa najmä pri šifrovaní menšieho množstva dát, pri digitálnych podpisoch, digitálnych certifikátoch a podobne.

Obr. 2.5: Rozdiel medzi symetrickým a asymetrickým šifrovaním

Pre navrhované riešenie je vhodnejšie použiť symetrický šifrovací algoritmus. Behom komunikácie medzi užívateľskou a serverovou časťou bude prebiehať výmena väčšieho množstva dát, preto je dôležité zaistiť okrem bezpečnosti aj čo najrýchlejší prenos.

Použitím symetrického šifrovania ale vzniká ďalší problém. Keďže sa pre šifrovanie aj dešifrovanie využíva jeden a ten istý šifrovací kľúč, je nutné, aby obe komunikujúce zariadenia tento šifrovací kľúč mali k dispozícii. Zároveň

musí byť šifrovací kľúč utajený. V prípade získania kľúča treťou stranou je takýto subjekt schopný komunikáciu bez problémov dešifrovať. Kľúč musí byť aj kvôli tomuto faktu pre každé spojenie nový a vygenerovaný náhodne. Problém, ktorý vzniká, je problém distribúcie šifrovacieho kľúča z jedného zariadenia na druhé. Jediným spôsobom komunikácie medzi zariadeniami je v navrhovanom riešení technológia Bluetooth. Je teda nutné navrhnúť taký spôsob výmeny šifrovacieho kľúča, ktorý neohrozí jeho utajenie a zároveň zabezpečí, že obe zariadenia získajú rovnaký kľúč.

2.5.3 Využitie asymetrického šifrovania

Možným riešením problému opísaného v predchádzajúcej časti je využitie asymetrického šifrovania pre distribúciu šifrovacieho kľúča. Zariadenie A v takom prípade odošle svoj verejný kľúč zariadeniu B, pomocou ktorého zariadenie B zašifruje šifrovací kľúč (vygenerovaný predtým). Takto zašifrovaný kľúč odošle naspäť zariadeniu A. To zašifrovaný šifrovací kľúč dešifruje. Napriek tomu, že sa takéto riešenie zdá byť funkčné, obnáša hneď niekoľko bezpečnostných hrozieb:

- generovanie šifrovacie kľúča na jednom zariadení,
- určenie rolí pri nadväzovaní komunikácie,
- overenie identity zariadení,
- odosielanie samotného šifrovacieho kľúča pomocou BT.

2.5.4 Inšpirácia protokolom TLS 1.3

Všetky bezpečnostné hrozby vymenované v predchádzajúcej časti musia byť počas návrhu adresované. V rámci zabezpečenia je ideálnejšie riešenie také získanie spoločného šifrovacieho kľúča, ktoré nevyžaduje jeho priame odosielanie. Pri návrhu takéhoto spôsobu som sa inšpiroval kryptografickým protokolom *Transport Layer Security* (skrátene *TLS*), ktorý je primárne určený pre Internet.

S vydaním *TLS 1.3* okrem mnohých vylepšení prišlo aj zjednodušenie nadväzovania bezpečnej komunikácie (nazývané termínom *handshake*). V rámci tohto procesu je pre výpočet spoločného šifrovacieho kľúča na oboch zariadeniach využívaný kryptografický protokol *Diffie-Hellman s využitím eliptických kriviek* (skrátene *ECDH* z anglického *Eliptic-curve Diffie-Hellman*). [19]

2.5.5 Diffie-Hellman

Pomocou ECDH je možné vypočítať spoločný šifrovací kľúč bez jeho priameho prenosu medzi zariadeniami. Namiesto toho obe zariadenia odosielajú verejnú

časť z dvojice hodnôt, ktorú si každé zariadenie vygeneruje. Jedinou podmienkou je výber rovnakej eliptickej krivky na oboch zariadeniach. Kombináciou svojej súkromnej časti a verejnej časti druhého zariadenia je na oboch zariadeniach vypočítaná hodnota, ktorú je následne možné použiť ako šifrovací kľúč. V implementácií mnou navrhnutého riešenia v neskorších kapitolách využívam eliptickú krivku *P-384*, ktorá je podľa [20] jednou z odporúčaných eliptických kriviek pre použitie vládou USA.

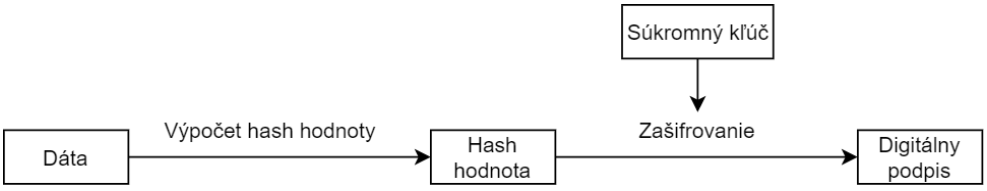
S využitím ECDH je odstránený problém prenosu samotného šifrovacieho kľúča cez kanál medzi zariadeniami. Vzniká však ďalšia bezpečnostná hrozba, ktorá sa týka overenia identity zariadení. Tento problém samotný ECDH nerieši, čím vzniká hrozba MITM útoku. Pri prijímaní verejných častí vygenerovaných dvojíc je nutné zaistiť, aby prijaté dáta patrili naozaj zariadeniu, s ktorým má byť spojenie nadviazané.

2.5.6 Overenie identity

Počas nadväzovania zabezpečeného spojenia medzi zariadeniami musí byť serverová časť aj užívateľská časť schopná overiť identitu druhej strany. Jedným z možných spôsobov je využitie BD_ADDR, unikátneho identifikátora, ktorý som predstavil už v predchádzajúcich častiach práce. Tento spôsob však nemožno plnohodnotne využiť z dôvodu, že útočník je schopný BD_ADDR na svojom zariadení zmeniť na adresu zariadenia, za ktoré sa v prípade MITM útoku má záujem vydávať.

Riešením tohto problému je použitie digitálnych certifikátov. Okrem iného sú digitálne certifikáty používané k digitálnym podpisom dát a následnej verifikácií týchto dát a vytvoreného digitálneho podpisu.

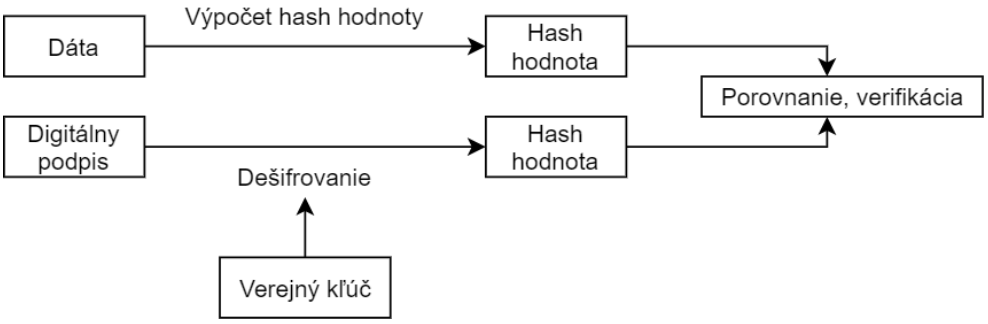
Princípom digitálnych podpisov je výpočet hash hodnoty podpisovaných dát a jej zašifrovanie súkromným kľúčom príslušného certifikátu. Tak je zaistené, že podpis vytvára iba taký subjekt, ktorý má k dispozícii súkromný kľúč príslušného certifikátu, ktorý ostáva utajený.



Obr. 2.6: Vytvorenie digitálneho podpisu

Táto zašifrovaná hodnota je finálny digitálny podpis dát. Verifikácia digitálneho podpisu prebieha podobne. Podpis je dešifrovaný verejným kľúčom certifikátu, ktorý digitálny podpis vytvoril. Tento kľúč možno verejne distribuovať kdekoľvek a komukoľvek. Výpočtom hash hodnoty dát, ktorých podpis

je overovaný, a následným porovnaním obidvoch hodnôt je nakoniec podpis verifikovaný. [21]



Obr. 2.7: Overenie digitálneho podpisu

Okrem toho musí byť každý digitálny certifikát digitálne podpísaný *certifikačnou autoritou* (skrátene *CA*). Certifikát môže byť podpísaný sám sebou (sám sebe je certifikačnou autoritou), avšak pre použitie v mojej práci je ideálne použiť ako certifikačnú autoritu druhý certifikát. Tento certifikát je určený len pre túto činnosť, čím sa z neho stáva dôveryhodná certifikačná autorita v rámci navrhovaného riešenia. [22]

Spoločným použitím digitálneho podpisu dát vytvoreného digitálnym certifikátom podpísaným dôveryhodnou CA je možné plnohodnotne overiť, že zariadenie, od ktorého sú dáta prijímané, nie je zariadenie útočníka.

Kapitola 3

Návrh

V tejto časti práce využívam informácie nazbierané v predchadzajúcich častiach práce a vytváram návrh riešenia, ktoré zabezpečí bezpečnú tlač dokumentov pomocou BT s využitím RPi počítača ako serverovej a Android zariadenia ako užívateľskej časti. Postupne opisujem technológie ktoré budú použité a užívateľské prostredie Android aplikácie.

3.1 Komunikácia medzi užívateľskou a serverovou časťou

Užívateľská a serverová časť budú komunikovať pomocou technológie Bluetooth.
Pre komunikáciu medzi zariadeniami bude použitý komunikačný protokol *RFCOMM*, ktorý ponúka jednoduchú a spoľahlivú implementáciu podobnú komunikačnému protokolu *TCP* určenému pre Internet.

V rámci správneho fungovania musí byť časť implementovaného riešenia, ktorá spravuje BT komunikáciu, identifikovaná pomocou *univerzálneho unikátneho identifikátora* (32 hexadecimálnych číslic, skrátene *UUID*). Tento identifikátor je využívaný pre rozpoznanie služieb, preto môže byť verejný a zároveň musí byť vo všetkých zariadeniach (figurujúcich ako užívateľské aj serverové časti) rovnaký.

Po nadviazaní zabezpečnéj komunikácie, ktorá je detailnejšie opísaná v ďalších častiach, bude celá komunikácia šifrovaná symetrickou šifrou *AES-256* v operačnom móde *Cipher Block Chaining* (skrátene *CBC*). Pre ošetrenie prípadu neaktivity užívateľskej časti bude možné na serverovej časti nastaviť čas, po uplynutí ktorého bude užívateľská časť automaticky odpojená, pokiaľ behom tejto doby neprijala serverová časť žiadne dáta.

3.1.1 Overenie identity

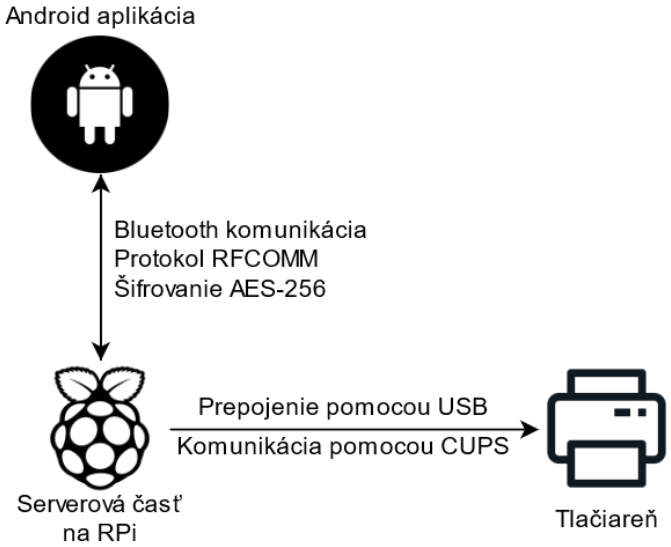
V predchádzajúcich kapitolách práce som zistil, že dôležitou bezpečnostnou súčasťou návrhu je schopnosť overenia identity užívateľskej aj serverovej časti. Pre túto činnosť budú v navrhnutom riešení používané dva spôsoby.

Pre overenie identity serverovej časti užívateľskou časťou bude využívaný princíp digitálnych certifikátov a digitálnych podpisov. Overenie identity užívateľskej časti serverovou časťou bude riešené databázou zariadení lokálne uloženou na serverovej časti. Pri zapnutom overovaní užívateľskej časti bude užívateľ vyzvaný, aby zadal heslo, ktoré bude porovnané s heslom uloženým v lokálnej databáze.

Oba typy overenia identity bude možné na serverovej časti nezávisle na sebe vypnúť a znova zapnúť.

3.2 Komunikácia medzi Raspberry Pi a tlačiarňou

Komunikácia medzi Raspberry Pi a tlačiarňou bude prebiehať pomocou USB kábla prepájajúceho tlačiareň a RPi. Týmto spôsobom bude zaistená kompatibilita s drvivou väčšinou súčasných aj starších tlačiarní. Pre komunikáciu na softvérovej úrovni Raspberry Pi bude používať systém *Common UNIX Printing System* (skrátene *CUPS*).



Obr. 3.1: Blokové schéma návrhu

3.3 Android aplikácia

Pre interakciu užívateľa s navrhnutým riešením slúži užívateľská časť zastúpená aplikáciou pre Android zariadenie. Pomocou aplikácie je schopný užívateľ nadviazať spojenie so spárovaným zariadením, na ktorom beží serverová časť. Aplikácia taktiež slúži pre všetky ostatné aspekty práce s navrhnutým riešením. Užívateľ je schopný v aplikácií zvoliť dokument uložený na Android zariadení. Ten následne po zvolení tlačiarne jedným kliknutím odošle na serverovú časť.

V prípade zapnutého overovania užívateľskej časti na serverovej časti je v aplikácií užívateľ vyzvaný k zadaniu hesla behom nadväzovania zabezpečeného spojenia. Okrem toho užívateľ do bezpečnosti komunikácie nemusí žiadnym spôsobom zasahovať, všetko prebieha autonómne na pozadí.

V prípade vypnutého používania serverového certifikátu je užívateľ po nadviazaní spojenia o tejto skutočnosti v aplikácií upozornený varovným textom. Rozhodnutie o pokračovaní práce s aplikáciou alebo o jej prerušení už je na samotnom užívateľovi.

Kapitola 4

Realizácia

V tejto časti práce dokumentujem realizáciu riešenia navrhnutého v predchádzajúcej časti práce. Postupne opisujem spôsob implementácie jednotlivých častí návrhu serverovej aj užívateľskej časti. Okrem iného opisujem použitie špecializovaných knižníc pre zabezpečenie komunikácie medzi oboma časťami.

4.1 Komunikácia medzi užívateľskou a serverovou časťou

V rámci mojej práce som zvolil pre UUID, unikátny identifikátor pomocou ktorého sa v rámci Bluetooth komunikácie bude riešenie identifikovať, hodnotu `17f3110c-7cfa-11eb-9439-0242ac130002`.

Pre overenie identity serverovej časti užívateľskou časťou je využívaný princíp digitálnych certifikátov a digitálnych podpisov. Na serverovej časti je uložený digitálny certifikát, ktorý je podpísaný certifikačnou autoritou. Certifikát je vytvorený pre danú serverovú časť, jeho položka *Common Name* obsahuje BT_ADDR zariadenia, na ktorom je serverová časť spustená. Digitálny certifikát CA je lokálne uložený na užívateľskej časti.

Overenie identity užívateľskej časti serverovou časťou je riešené databázou zariadení lokálne uloženou na serverovej časti. Databáza obsahuje pre každé zariadenie jeho BT_ADDR, hash hodnotu hesla, ktoré užívateľ zadáva behom nadväzovania zabezpečeného spojenia a soľ, ktorá bola použitá pri výpočte hash hodnoty hesla. Hodnota soli je pre každé zariadenie náhodne generovaná. Pri výpočte hash hodnoty je soľ pridaná na začiatok ukladaného hesla, pričom až táto hodnota je predávaná hash funkcií *SHA-256*. Hodnota vypočítaná funkciou je následne uložená do databázy.

4.1.1 Nadviazanie komunikácie

Pri nadväzovaní komunikácie sa vždy užívateľská časť (Android zariadenie) pripája k serverovej časti. Prvým krokom je spárovanie zariadení cez BT. Spárovanie prebieha cez BT nastavenie na Android zariadení a je úplne nezávislé od implementácie mnou navrhnutého riešenia.

Po spárovaní zariadení sa užívateľ pripája pomocou užívateľskej aplikácie, ktorá je detailnejšie popísaná v poslednej časti tejto kapitoly. Android zariadenie sa pripája na spárované zariadenie, na ktorom je spustená serverová časť. Serverová časť každé takéto pripojenie príjme. V prípade zapnutého overovania identity užívateľskej časti serverová časť overi prítomnosť BT_ADDR príslušného zariadenia v lokálnej databáze. V prípade, že táto adresa v databáze nefiguruje, pripojenie ukončí.

Po vzniku pripojenia sa zaháji proces výpočtu spoločného šifrovacieho kľúča na oboch zariadeniach. Tento proces je zastúpený výpočtom spoločnej hodnoty pomocou ECDH s použitím eliptickej krivky P-384 a následným výpočtom hash hodnoty funkciou SHA-256.

Obe zariadenia vygenerujú verejnú a súkromnú časť pre výpočet spoločnej hodnoty. Ako prvé odošle Android zariadenie **Klient paket**. Jedná sa o jeho verejnú časť veľkosti 120 bajtov. RPi zariadenie paket príjme a kombináciou so svojou súkromnou časťou vypočíta spoločnú hodnotu. Z tejto hodnoty je pomocou hashovacej funkcie SHA-256 vypočítaná hash hodnota, ktorá je v následnej komunikácii používaná ako šifrovací kľúč. Následne RPi zariadenie odošle **Server paket**, ktorý obsahuje až 6 položiek.

Obr. 4.1: Schéma Server paketu

Keyshare je verejná časť využívaná pre výpočet spoločnej hodnoty pomocou ECDH. Rovnako ako u užívateľskej časti sa jedná o 120 bajtov informácie.

CertSize je číslo (4 bajty informácie) určujúce veľkosť serverového certifikátu, ktorý je využívaný pre overenie identity serverovej časti. Táto položka môže byť nulová v prípade, že overovanie serverovej časti je vypnuté.

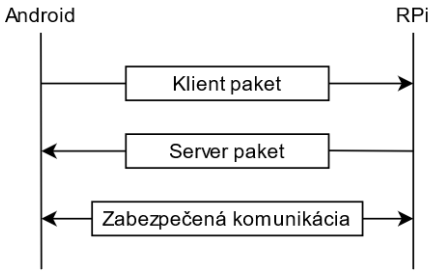
PwdTries označuje číslo (4 bajty informácie) určujúce maximálny počet zadaní nesprávneho hesla pri overovaní užívateľskej časti. Táto hodnota môže byť ľubovoľne nastavená na serverovej časti. V prípade vypnutého overovania užívateľskej časti je táto hodnota nulová.

Nasledujúce položky sú odoslané v závislosti na zapnutých overovaniach identít na serverovej časti.

V prípade, že je overovanie serverovej časti certifikátom zapnuté, je štvrtou položkou **digitálny podpis keyshare-u**, ktorý bol odoslaný na začiatku paketu. Táto položka má 256 bajtov.

Nasleduje **digitálny certifikát**, ktorým bol podpis v predchádzajúcej časti vytvorený. Jeho veľkosť bola zaslaná už predtým v položke *CertSize*.

V prípade, že je overovanie užívateľskej časti zapnuté, je poslednou položkou **Salt**. Jedná sa o soľ, ktorá bola použitá pri výpočte hash hodnoty užívateľského hesla uloženého v databáze na serverovej časti. Táto položka je potrebná pre následné zasielanie hash hodnoty hesla (zadaného užívateľom v aplikácii) z užívateľskej na serverovú časť.



Obr. 4.2: Nadviazanie komunikácie

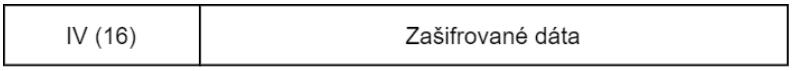
Po prijatí Server paketu užívateľskou časťou je aplikáciou overená platnosť prijatého certifikátu a podpis certifikačnej autority. Následne je overený digitálny podpis keyshare-u. V prípade zlyhania ktoréhokoľvek kroku je pripojenie ukončené. V opačnom prípade je taktiež vypočítaná spoločná hodnota, ktorá je totožná s vypočítanou hodnotou na serverovej časti. Z tejto hodnoty je nakoniec vypočítaná hash hodnota hashovaciou funkciou SHA-256. Táto hash hodnota je rovnaká ako na serverovej časti a stáva spoločným šifrovacím kľúčom. Následná komunikácia je už symetricky šifrovaná šifrou AES-256 za pomoci vypočítaného šifrovacieho kľúča.

4.1.2 Výmena dát

Celá výmena dát po nadviazaní zabezpečeného spojenia je šifrovaná. Keďže je pre šifrovanie využívaná bloková symetrická šifra AES-256 v operačnom móde CBC, je potrebné dáta pred zašifrovaním doplniť do veľkosti deliteľnej veľkosťou šifrového bloku, ktorá je pre AES 16 bajtov. Pre výpĺň dát sa využíva výpĺň typu *PKCS7*, ktorá vždy doplní do dát aspoň 1 bajt. To znamená, že pri dátach, ktorých veľkosť už je deliteľná číslom 16, pridá výplň *PKCS7* ďalších 16 bajtov.

Okrem výplne je nutné pre každé šifrovanie generovať *inicializačný vektor* (skrátene *IV*). Jedná sa o 1 šifrový blok (16 bajtov) náhodne vygenerovaných hodnôt, ktorý je použitý na začiatku šifrovania. Inicializačný vektor je nutné posielať spoločne so zašifrovanými dátami.

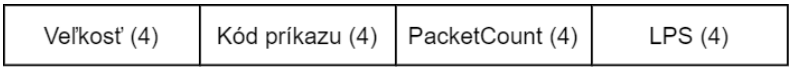
Android a RPi zariadenie medzi sebou komunikujú pomocou *dátových paketov*. Jedná sa o dátové bloky veľké maximálne 1024 bajtov.



Obr. 4.3: Schéma dátového paketu

Dátový paket začína blokom 16 bajtov, ktorý tvorí inicializačný vektor. Tento inicializačný vektor bol pri šifrovaní použitý na zašifrovanie dát, ktoré tvoria zbytok príslušného dátového paketu (maximálne teda 1008 bajtov). Z dôvodu používania výplne *PKCS7* je možné do jedného dátového paketu zašifrovať maximálne 992 bajtov dát. Spoločne s výplňou, ktorá v tomto prípade tvorí 16 bajtov, a inicializačným vektorom, ďalších 16 bajtov, získame dátový paket maximálnej veľkosti 1024 bajtov.

Prvým odosielaným dátovým paketom je *príkazový paket*. Po jeho dešifrovaní získame štruktúru, ktorá obsahuje 4 položky.



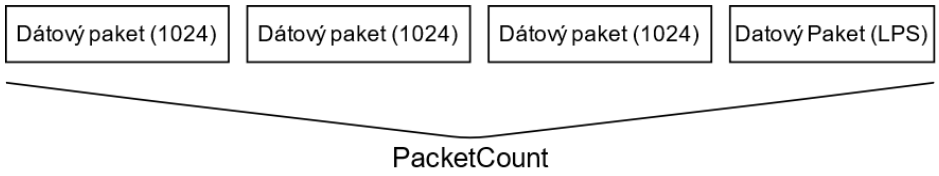
Obr. 4.4: Schéma príkazového paketu

Prvou položkou je **veľkosť zašifrovaného príkazového paketu**. Keďže je príkazový paket fixnej veľkosti, využíva sa táto položka ako kontrolná hodnota. Jej hodnota by mala byť vždy nastavená na 48.

Nasleduje ďalšie číslo, ktoré reprezentuje **kód príkazu**. V rámci komunikácie medzi užívateľskou a serverovou časťou je definovaných niekoľko kódov príkazov, ktoré určujú, čo zariadenia medzi sebou chcú vykonávať (odosielanie hash hodnoty hesla, odosielanie dát pre tlač a podobne).

Treťou položkou je číslo **PacketCount**, ktoré určuje počet nasledujúcich dátových paketov, ktoré k danému príkazu patria. Jeho hodnota môže byť nulová, ak je príkazový paket jediný paket daného príkazu.

Poslednou položkou je číslo **LPS**, ktoré určuje veľkosť posledného dátového paketu.



Obr. 4.5: Dátové pakety odoslané po príkazovom pakete

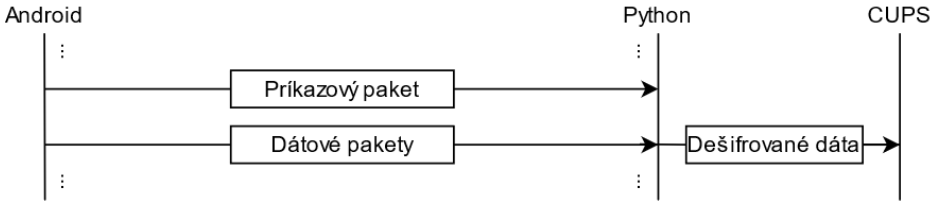
Príkazový paket obsahuje informácie o príkaze a celkový počet a veľkosť dátových paketov odoslaných po príkazovom pakete.

Špeciálnym prípadom výmeny dát je odosielanie dokumentov pre tlač. Pri tomto príkaze obsahuje prvý dátový paket odoslaný po príkazovom pakete ako prvé 4 bajty číslo, ktoré označuje poradové číslo tlačiarne nakonfigurovanej na serverovej časti. Za týmto číslom už nasledujú dáta dokumentu, ktorý je odosielaný pre tlač.

4.2 Komunikácia medzi Raspberry Pi a tlačiarňou

Pre komunikáciu medzi RPi a tlačiarňou je na softvérovej úrovni použitý systém CUPS. Jedná sa o jeden z najpoužívanejších systémov pre komunikáciu s tlačiarňou a tlač dokumentov pre Linux distribúcie. Ponúka jednoduchú správu tlačiarní a tlačových úloh pomocou príkazového riadka alebo webového rozhrania. Okrem iného ponúka aj možnosť zdieľania tlačiarní pripojených k zariadeniu pomocou USB kábla na sieti. Vďaka tomu je možné z USB tlačiarne vytvoriť jednoduchým a elegantným spôsobom tlačiareň sieťovú.

Vďaka rozsiahlemu verejne dostupnému API je možné využiť CUPS v rámci mojej práce. Po prijatí príslušného príkazového paketu označujúceho tlač dokumentov od užívateľskej časti príjme prvý nasledujúci dátový paket. Po načítaní identifikačného čísla tlačiarne vytvorí serverová časť v systéme CUPS novú tlačovú úlohu pre danú tlačiareň a odošle do nej zbytok prijatého dátového paketu. Následne postupne prijíma ďalšie dátové pakety. Každý dátový paket dešifruje a ihneď odosiela do systému CUPS do príslušnej tlačovej úlohy. Po prijatí všetkých dátových paketov odošle serverová časť príkaz na tlač. Systém CUPS odošle tlačovú úlohu do tlačiarne, ktorá prijaté dáta vytlačí.



Obr. 4.6: Prenos dát do systému CUPS

4.3 Serverová časť

Pre implementáciu serverovej časti som si vybral programovací jazyk *Python*. Vďaka jeho jednoduchému použitiu a rozsiahlemu množstvu modulov umožňuje implementáciu všetkých dôležitých súčastí, ktoré bude serverová časť spravovať. V nasledujúcich častiach okrem implementácie logiky serverovej časti opisujem aj správu nastavení a správu užívateľských zariadení.

4.3.1 Python

V programovacom jazyku Python som implementoval hlavnú logiku serverovej časti navrhnutého riešenia. Jedná sa okrem iného aj o komunikáciu s užívateľskou časťou pomocou BT a komunikáciu s tlačiarňou pomocou CUPS. Avšak najdôležitejšou logikou serverovej časti je samotné zabezpečenie komunikácie medzi zariadeniami pomocou BT.

Pre zabezpečenie komunikácie som využil triedy z modulu `cryptography`. Jedná sa o rozsiahly modul pre programovací jazyk Python, ktorý ponúka všetky dôležité bezpečnostné prvky využívané na serverovej časti.

Pre výpočet hash hodnoty pomocou SHA-256 som implementoval jednoduchú triedu `Sha256` nachádzajúcu sa v súbore `sha.py`. Obsahuje jedinú metódu, ktorá z prijatých dát vypočíta príslušnú hash hodnotu.

Pre správne a jednoduché používanie prvkov v rámci ECDH som vytvoril triedu `EllipticDiffieHellman`, ktorá je implementovaná v súbore `edh.py`. Táto trieda umožňuje generovanie súkromnej aj verejnej časti, načítanie verejnej časti druhého zariadenia a výpočet spoločného šifrovacieho kľúča. Pri výpočte volá triedu `Sha256`, ktorú som opísal v predchádzajúcom odstavci.

O šifrovanie pomocou AES-256 v operačnom móde CBC s výplňou typu PKCS7 sa stará trieda `Aes256` implementovaná v súbore `aes.py`. V tejto triede využívam okrem modulu `cryptography` aj modul `os` a jeho metódu `urandom`, ktorá umožňuje bezpečné generovanie náhodných hodnôt danej veľkosti. Túto metódu využívam pri generovaní IV pri šifrovaní dát.

Pre jednoduchú prácu s digitálnymi certifikátmi a vytváranie digitálnych podpisov som vytvoril sadu pomocných funkcií, ktoré sú obsiahnuté v súbore `certificate.py`.

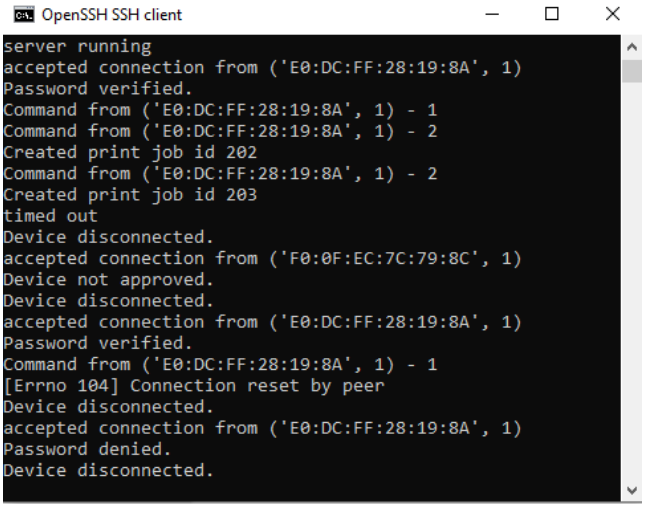
Komunikáciu s tlačiarňou cez systém CUPS som implementoval v triede `CupsWrapper`. Trieda sa nachádza v súbore `cups_wrapper.py` a používa modul `pycups`. Jedná sa o modul, ktorého metódy interne volajú originálne API systému CUPS napísané v programovacom jazyku C. Vďaka tomuto API je možné jednoduchým spôsobom komunikovať so systémom CUPS, načítať z neho nainštalované tlačiarne a vytvárať tlačové úlohy.

Práca s lokálnou databázou je docielená pomocou triedy `DatabaseWrapper`, ktorú som implementoval pomocou modulu `sqlite3`. Implementácia je uložená v súbore `database.py` a jej chovanie, rovnako ako štruktúra databázy je popísaná v ďalších častiach práce.

Všetky vyššie opísané triedy sú importované v triede `BluetoothServer`, ktorá obsahuje hlavnú logiku serverovej časti. Okrem toho táto trieda využíva modul `bluetooth`, ktorý obsahuje logiku pre komunikáciu zariadení pomocou BT. Pre načítanie konfiguračného súboru, ktorého štruktúra a fungovanie je opísané neskôr, využíva trieda taktiež modul `yaml`.

Serverová časť sa spúšťa pomocou bash skriptu `btcups-run`. Z dôvodu práce s BT je nutné tento skript spúšťať ako superuser. Na túto skutočnosť aj skript upozorňuje v prípade, že nebol spustený ako superuser. Po spustení

skript spustí pomocný Python skript `run.py`, ktorý vytvorí inštanciu triedy `BluetoothServer` a pomocou jej metód spravuje beh serverovej časti.



Obr. 4.7: Príklad výstupu behu serverovej časti

4.3.2 Úprava nastavení

Pre jednoduchú úpravu nastavení serverovej časti som oddelil základné premenné mimo zdrojové kódy do konfiguračného súboru `config.yaml`. Konfiguračný súbor obsahuje premenné, ktoré vplývajú na chod a správanie serverovej časti.

```
APP:
    TIMEOUT: 60
    MAX_TOTAL_DATA: 10485760
    UUID: 17f3110c-7cfa-11eb-9439-0242ac130002

    PASSWORD_VERIFICATION: True
    PASSWORD_MAX_TRIES: 3

    CERT_USAGE: True

CERTIFICATE:
    CERT_PATH: cert/cert.crt
    KEY_PATH: cert/private.key

DATABASE:
    DATABASE_PATH: database.db
```

Konfiguračný súbor je rozdelený na tri základné kategórie.

V kategórií `APP` sú tie premenné, ktoré sú načítavané pri každom spustení serverovej časti nezávisle na jej nastavení.

`TIMEOUT` označuje čas v sekundách, po ktorom je užívateľská časť odpojená, pokiaľ neodoslala na serverovú časť žiadne dáta. Jej hodnotu možno ľubovoľne meniť.

`MAX_TOTAL_DATA` označuje maximálnu veľkosť, ktorú serverová časť od užívateľskej časti akceptuje. Táto hodnota je kontrolovaná po prijatí príkazového paketu a pred prijímaním ďalších dátových paketov. Túto hodnotu možno taktiež ľubovoľne meniť.

`UUID` označuje univerzálny unikátny identifikátor, ktorým sa serverová časť identifikuje v rámci BT v čase, kedy nekomunikuje so žiadnym užívateľským zariadením. Túto hodnotu možno zmeniť, avšak pre správne fungovanie je v takom prípade nutné UUID zmeniť aj na všetkých užívateľských zariadeniach, ktoré sa k danej serverovej časti chcú v budúcnosti pripojiť.

Pomocou `PASSWORD_VERIFICATION` je možné ovládať overovanie identity užívateľskej časti. V prípade zapnutého overovania serverová časť načíta informácie z kategórie `DATABASE`. Jedinou informáciou v tejto kategórii je cesta k databázovému súboru, ktorá je uložená v premennej `DATABASE_PATH`.

Hodnota `PASSWORD_MAX_TRIES` určuje maximálny povolený počet neúspešných pokusov o zadanie hesla pri nadväzovaní zabezpečeného spojenia. Túto hodnotu možno ľubovoľne meniť.

Pomocou `CERT_USAGE` je možné ovládať overovanie identity serverovej časti pomocou digitálneho certifikátu a digitálnych podpisov. Pri zapnutom overovaní serverová časť načíta informácie z kategórie `CERTIFICATE`, ktorá obsahuje cesty k serverovému certifikátu (premenna `CERT_PATH`) a jeho súkromnému kľúču (`KEY_PATH`), ktorý je využívaný pre vytvorenie digitálneho podpisu.

4.3.3 Správa užívateľských zariadení

Pre správu užívateľských zariadení a ich kontrolu v prípade zapnutého overovania identity užívateľskej časti je nutné na serverovej časti implementovať lokálnu databázu. Túto databázu som už opisoval v predchádzajúcej kapitole. Pre jej implementáciu som si zvolil relačnú databázu typu *SQLite*. Na rozdiel od ostatných typov relačných databáz je možné tento typ nasadiť rýchlo a jednoducho pomocou jediného databázového súboru. Medzi ďalšie výhody patrí modul pre programovací jazyk Python, ktorý umožňuje jednoduchú správu takejto databázy.

Databáza obsahuje jedinú tabuľku `devices`. V nej sú obsiahnuté stĺpce `id`, `addr`, `pwd` a `salt`. Jeden riadok v tabuľke predstavuje jedno užívateľské zariadenie, ku ktorému je ukladaná jeho BT_ADDR, hash hodnota hesla a soľ, ktorá bola pri výpočte hash hodnoty hesla použitá.

Pre správu (pridávanie zariadení, zmena hesla) zariadení som implementoval pomocný skript `btcups-admin`, ktorý interne spúšťa súbor `admin.py`

naprogramovaný v jazyku Python. Skript obsahuje jeden argument, cestu k databázovému súboru. Po jeho spustení prebehne sken viditeľných BT zariadení v okolí pomocou modulu `bluetooth`. Po dokončení skenovania skript ponúkne výber z naskenovaných zariadení, prípadne možnosť zadať BT_ADDR zariadenia ručne. Následne vyžiada zadanie hesla. Skript automaticky vygeneruje novú náhodnú hodnotu soli a vypočíta hash hodnotu hesla, ktorú uloží spoločne s adresou do databáze. V prípade, že už zariadenie so zadanou BT_ADDR v databáze existuje, príslušný riadok aktualizuje.

4.3.4 Inštalačný skript

Pre jednoduchšie nasadenie serverovej časti na RPi som vytvoril bash skript `install`, ktorý umožňuje pripraviť celú serverovú časť pomocou jedného príkazu. Skript automaticky nainštaluje všetky knižnice, prerekvizity a Python moduly využívané v implementácií navrhnutého riešenia. Taktiež upraví a nastaví všetky potrebné interné nastavenia. Ako posledné vytvorí databázový súbor `database.db`, v ktorom vytvorí prázdnu tabuľku podľa štruktúry opísanej v predchádzajúcej časti.

Skript je určený pre *Raspberry Pi OS*. Keďže inštaluje globálne knižnice a upravuje interné konfiguračné súbory operačného systému, je nutné ho spúšťať ako superuser (`sudo ./install`).

Inštalačný skript sa pokúsi v prvom rade o inštaláciu systému CUPS. V prípade, že systém CUPS na zariadení nie je nainštalovaný a nie je v ňom tlačiareň nakonfigurovaná, je nutné pred začiatkom používania implementovaného riešenia tlačiareň nakonfigurovať.

To je možné docieliť jednoduchým spôsobom cez webové rozhranie, ktoré je možno aktivovať príkazom `sudo cupsctl --remote-any --remote-admin`. Webového rozhranie je dostupné cez internetový prehliadač na adrese `[IP adresa RPi]:631/`. Pre možnosť inštalácie novej tlačiarne je ešte nutné pridať užívateľa do skupiny `lpadmin`. Pre užívateľa `pi` by bolo pridanie do danej skupiny docielené napriklad príkazom `sudo gpasswd -a pi lpadmin`. Po dokončení inštalácie by malo byť z bezpečnostných dôvodov rozhranie deaktivované príkazom `sudo cupsctl --no-remote-any --no-remote-admin`.

V prípade, že je inštalačný skript spúšťaný na čerstvej verzii operačného systému, je nutné pred jeho spustením systém aktualizovať pomocou príkazov `sudo apt-get update` a `sudo apt-get upgrade`.

4.4 Užívateľská časť

Pre implementáciu užívateľskej aplikácie pre Android zariadenia, ktorú som nazval *BtCups*, som si vybral programovací jazyk Kotlin. Tento programovací jazyk ponúka možnosť využívať interné knižnice programovacieho jazyka Java, v ktorom je napísané API operačného systému Android. Kotlin som si

vybral hlavne z dôvodu modernej syntaxe, rýchlosti a podobnej funkcionality ako programovací jazyk Python.

4.4.1 Kotlin

Podobne ako pri serverovej časti, dôležitou súčasťou implementovanej užívateľskej časti sú triedy a metódy umožňujúce zabezpečenú komunikáciu so serverovou časťou.

Výpočet hash hodnôt pomocou hashovacej funkcie SHA-256 som implementoval v trieda `Sha256`. Táto trieda má jedinú metódu, ktorá vypočíta hash hodnotu dodaných dát. K tomu využíva triedu `MessageDigest` z Java knižnice `java.security`.

Celý proces ECDH som implementoval triedou `EllipticDiffieHellman`. Táto trieda umožňuje generovanie verejnej a súkromnej časti, načítanie verejnej časti od druhého zariadenia a výpočet spoločného šifrovacieho kľúča. To je docielené triedami z Java knižníc `java.security` a `java.crypto`. Okrem toho je využívaná taktiež trieda `Sha256`, ktorú som opísal v predchádzajúcom odstavci.

Šifrovanie pomocou šifrovacieho algoritmu AES-256 v operačnom móde CBC s výplňou typu PKCS7 som implementoval v triede `Aes256`. V tejto triede sa využívajú triedy z Java knižnice `java.crypto`, ale taktiež z Kotlin knižnice `kotlin.random`, vďaka ktorej sa generujú inicializačné vektory pri šifrovaní dát. Zaujímavosťou je, že pri inicializácii šifry AES-256 v operačnom móde CBC poskytuje Java knižnica `java.crypto` výplň typu `PKCS5Padding`, ktorá je avšak implementovaná ako výplň typu PKCS7.

Prácu s certifikátmi som implementoval v triede `CertificateWrapper`. Oproti serverovej časti je táto trieda rozsiahlejšia, keďže okrem načítania certifikátov taktiež certifikáty overuje. Overuje, či sú digitálne podpísane certifikačnou autoritou a overuje digitálne podpisy dát, ktoré certifikát vytvoril. K tomu využívam triedy z Java knižnice `java.security`.

Všetky vyššie opísané triedy využíva trieda `BluetoothWrapper`, do ktorej som implementoval celú logiku Bluetooth komunikácie medzi užívateľskou a serverovou časťou riešenia. To som dosiahol použitím Android knižnice `android.bluetooth`.

Aplikácia pri spustení kontroluje stav BT na Android zariadení. V prípade vypnutého BT vyzve užívateľa k jeho zapnutiu. Ak užívateľ odmietne, aplikácia sa síce spustí, ale nie je použiteľná. Užívateľa upozorní na nutnosť spusteného BT. Pre jej správne fungovanie je nutné okrem zapnutia BT aplikáciu reštartovať.

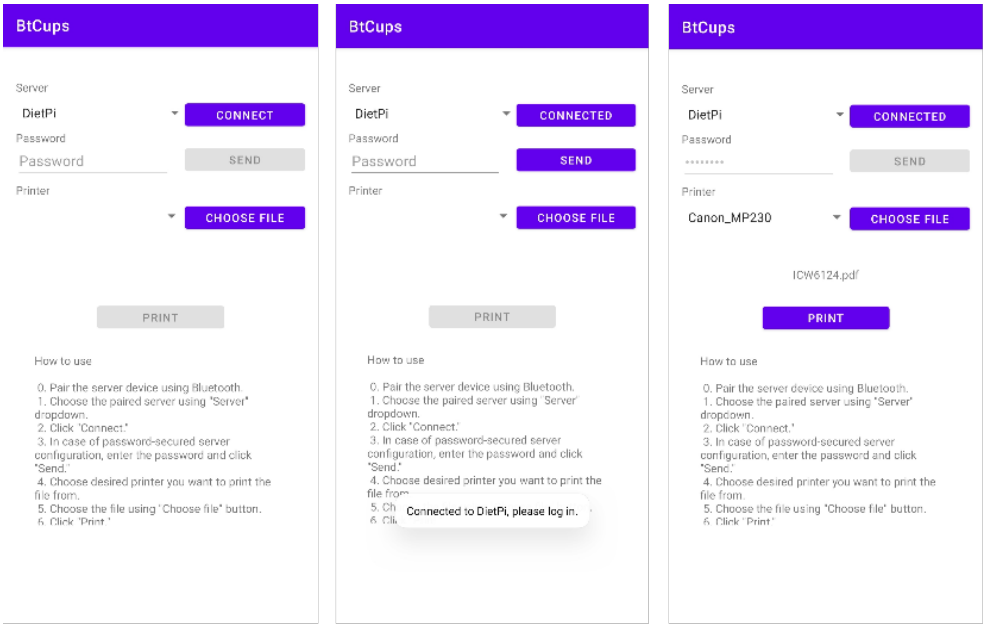
Obrazovka, ktorá je zobrazená užívateľovi po úspešnom spustení aplikácie, je implementovaná jedinou aktivitou `MainActivity`. Táto obrazovka obsahuje sekciu pre nadviazanie spojenia so serverovou časťou. K tomu slúži ponuka *Server* spoločne s tlačidlom *Connect*. Po stlačení tohto tlačidla je vytvorená inštancia triedy `BluetoothWrapper`, ktorá sa pokúsi nadviazať spo-

jenie. Po úspešnom nadviazaní spojenia sa podľa nastavení serverovej časti aktivujú ďalšie prvky v aktivite.

V prípade zapnutého overovania užívateľskej časti pomocou hesla je po nadviazaní spojenia aktivované textové pole *Password* spoločne s tlačídlom *Send*. Užívateľská časť po zadaní hesla do textového poľa a stlačení tlačidla vypočíta hash hodnotu pomocou prijatej hodnoty soli od serverovej časti a užívateľom zadaného hesla. Túto hodnotu odošle na serverovú časť a počká na overenie. Po úspešnom overení je sekcia pre zadávanie hesla deaktivovaná.

V prípade vypnutého overovania serverového certifikátu je v užívateľskej aplikácií zobrazená varovná správa o tejto skutočnosti. Aplikácia je funkčná a spojenie neukončuje, toto rozhodnutie ostáva na užívateľovi.

Po dokončení všetkých overovaní si užívateľská časť vyžiada zoznam tlačiarní nainštalovaných na serverovej časti. Po ich získaní naplní ponuku *Printers*. Užívateľ je schopný v akomkoľvek okamihu vyberať súbory pre tlač pomocou tlačidla *Choose File*. Toto tlačidlo je aktivované neustále, jeho aktivácia nezávisí od pripojenia na serverovú časť. V prípade, že je užívateľská časť pripojená a je zvolený súbor pre tlač, je aktivované tlačidlo *Print*. Po stlačení tohto tlačidla zvolený súbor odoslaný na serverovú časť a následne do tlačiarne.



Obr. 4.8: Postupné pripojenie a prihlásenie v užívateľskej aplikácii

Okrem vyššie zmienených obsahuje užívateľská časť aj stručný návod na použitie.

Kapitola 5

Testovanie

Posledná kapitola mojej práce obsahuje dokumentáciu a opis testovania implementovaného riešenia pre tlač dokumentov pomocou technológie Bluetooth. Testovaním som overil správne fungovanie a zistil doplňujúce informácie k mojej implementácii.

5.1 Testovacie prostredie

Pre testovanie som vybral ako serverovú časť *RPi Zero W*, pre ktoré som aj celé riešenie navrhoval a implementoval. Je však dôležité poznamenať, že napriek návrhu pre *RPi Zero W*, by malo finálne implementované riešenie fungovať na akejkoľvek hardwarovej platforme s distribúciou Linux a podporou BT.

Ako operačný systém som pre účely testovania zvolil operačný systém *Raspberry Pi OS*, ktorý som predstavil a analyzoval už v predchádzajúcich kapitolách práce. Keďže sa jedná o operačný systém vyvíjaný priamo výrobcom RPi, považujem zaručenie správneho fungovania konečného riešenia na tomto operačnom systéme za dôležitú súčasť práce.

Digitálny certifikát CA a príslušný súkromný kľúč, ktorý bol použitý pre podpis certifikátu pre serverovú časť, sú uložené v prílohe práce. Certifikát CA bol pre zaistenie správnej konfigurácie behom testovania lokálne uložený v užívateľskej aplikácií.

5.2 Inštalácia serverovej časti

Ako prvý test som vybral inštaláciu serverovej časti pomocou pripraveného skriptu. Po spustení skriptu pod štandardným užívateľom skript serverovú časť nepripravil, keďže nebol spustený ako superuser. Túto skutočnosť oznámil v termináli. Po spustení skriptu ako superuser (`sudo ./install`) skript úspešne nainštaloval a pripravil všetky komponenty nutné pre beh serverovej časti. Po pridaní zariadenia pomocou `sudo ./btcups-admin` a násled-

ného spustenia serverovej časti pomocou `sudo ./btcups-run` sa serverová časť úspešne spustila a bola schopná nadviazať spojenie s pridaným užívateľským zariadením.

5.3 Úprava nastavení serverovej časti

Druhým testom implementovaného riešenia bol test funkčného správania serverovej časti pri úprave konfiguračného súboru `config.yaml`.

5.3.1 Vypnutie serverového certifikátu

Po nastavení hodnoty `CERT_USAGE` na `False` a reštartovaní serverovej časti prebehlo načítanie konfiguračného súboru bez chyby. Pri následnom pripojení užívateľskej časti serverová časť správne nastavila položku *CertSize* v Server pakete na nulovú hodnotu a žiadny certifikát neodoslala. V užívateľskej aplikácií sa po vytvorení spojenia správne objavila varovná správa o nepoužívaní certifikátu. Funkcie odosielania dát a tlače dokumentov boli plne funkčné.

5.3.2 Vypnutie overovania identity užívateľa

Po nastavení `PASSWORD_VERIFICATION` v konfiguračnom súbore na `False` a reštartovaní serverovej časti prebehlo spustenie bez chyby. Pri následnom pripájaní užívateľských zariadení serverová časť správne odosielala položku *PwdTries* v Server pakete ako nulovú. Užívateľská aplikácia túto položku prijala a správne nežiadala od užívateľa zadanie hesla.

5.4 Viacero zapojených tlačiarní

V rámci správneho fungovania komunikácie medzi serverovou časťou a systémom CUPS som do systému CUPS na *RPi Zero W* nainštaloval druhú tlačiareň. Spustená serverová časť riešenia tlačiareň detekovala okamžite a pri najbližšom pripojení Android zariadenia odoslala informácie o oboch tlačiarňach. Užívateľská časť informácie prevzala a správne rozdelila na dve možnosti, ktorými naplnila ponuku *Printers* v aplikácii.

Užívateľská časť pri odosielaní dát na serverovú časť správne detekovala identifikačné číslo zvolenej tlačiarne, ktoré odoslala spoločne s odosielaným dokumentom. Aj po opakovaných testoch a rôznych výberoch oboch tlačiarní užívateľská časť správne odoslala identifikačné čísla tlačiarní. Serverová časť vždy identifikačné číslo správne prevzala a následné prijaté dáta odoslala do príslušnej tlačiarne.

5.5 Rýchlosť odosielania dokumentov

Jedným z dôležitých aspektov implementovaného riešenia je schopnosť a rýchlosť odosielania väčšieho množstva dát z užívateľskej časti na serverovú. Práve pri tlači dokumentov dochádza k prenosu najväčšieho množstva dát. V rámci tejto časti testovania som na serverovú časť odosielal pre tlač dokumenty rôznych veľkostí a zaznamenával čas potrebný na vytvorenie tlačovej úlohy v systéme CUPS. Čas potrebný pre celkovú tlač som už nezaznamenával, keďže táto premenná je závislá od použitej tlačiarne a je nezávislá od serverovej časti. V tabuľke nižšie je možné vidieť jednotlivé veľkosti dokumentov a čas potrebný na ich prenos z užívateľskej časti na serverovú časť implementovaného riešenia.

Veľkosť	Doba prenosu
120KB	1.5 sekúnd
280KB	2.6 sekúnd
500KB	4.5 sekúnd
1MB	9.2 sekúnd
2MB	19 sekúnd
5MB	59.6 sekúnd

Tabuľka 5.1: Doba prenosu dokumentov rôznej veľkosti.

5.6 Dlhodobý beh serverovej časti

Behom tohto testu bola serverová časť implementovaného riešenia spustená na *RPi Zero W* nepretržite po dobu 10 dní. Behom tohto obdobia boli na serverovú časť pripájané viaceré Android zariadenia. Všetky Android zariadenia sa úspešne pripojili, overili a boli schopné tlačiť dokumenty.

5.7 Odosielanie chybných dát

Nasledujúci test sa vzťahuje na správne zachytávanie chýb pri práci s dátami na serverovej časti. Pomocou upravenej verzie užívateľskej aplikácie som sa pripojil na serverovú časť a odosielal chybné dáta.

Prvým typom chybných dát boli dátové pakety s chybným inicializačným vektorom. Serverová časť dáta prijala a pokúsila sa ich dešifrovať, avšak triedy modulu `cryptography` identifikovali chybný inicializačný vektor a vyvolali chybu. Serverová časť túto chybu zaznamenala a ukončila s užívateľskou časťou spojenie.

Ako druhý typ chybných dát som zvolil nesprávne nastavené položky v príkazovom pakete. Pri nesprávnom nastavení prvej položky reprezentujúcej veľ-

kosť zašifrovaného príkazového paketu serverová časť okamžite ukončila spojenie. Po prijatí príkazového paketu s väčším počtom očakávaných dátových paketov po príkazovom pakete serverová časť neukončila spojenie a očakávala tieto dátové pakety. Po uplynutí doby nastavenej v konfiguračnom súbore premennou `TIMEOUT` serverová časť správne ukončila spojenie.

Ako posledný prípad chybných dát som zvolil výber súborov nesprávneho formátu v užívateľskej aplikácií. Po odoslaní dokumentov nespôsobilých pre tlač serverová časť tieto dáta prijala a poslala do systému CUPS. Ten sám vyhodnotil chybu a dáta zahodil pred odoslaním do tlačiarne.

5.8 Chybný certifikát

Správne fungovanie pri zapnutom a vypnutom používaní certifikátu na serverovej časti som už testoval. Pri tomto teste ostalo používanie certifikátu zapnuté, avšak na serverovú časť som nahral chybný certifikát, ktorý nie je podpísaný certifikačnou autoritou uloženou lokálne v užívateľskej aplikácií. Pri pokuse o pripojenie užívateľská aplikácia prijala Server paket spoločne s certifikátom od serverovej časti. Následne po neúspešnej verifikácii certifikátu voči lokálne uloženému certifikátu CA správne spojenie prerušila a aplikácia vypísala chybu o neúspešnom pripojení.

Po vypnutí používania certifikátu na serverovej časti Server paket už behom nadväzovania spojenia certifikát neobsahoval. Užívateľská časť rovnako ako pri predchádzajúcom teste certifikát nenačítala, ani sa nesnažila o jeho verifikáciu. V aplikácií sa taktiež znova zobrazila varovná správa o nepoužívaní serverového certifikátu. Užívateľská aj serverová časť nadviazali spojenie a funkcia odosielania dát aj tlače dokumentov boli plne funkčné.

5.9 Vypnutý Bluetooth na Android zariadení

Posledným testom bol test správania sa užívateľskej aplikácie pri vypnutom BT. Po spustení aplikácie sa správne objavilo okno so žiadosťou o jeho zapnutie. Po prijatí žiadosti aplikácia správne načítala spárované zariadenia, ktoré boli následne načítané do ponuky *Server*. Aplikácia bola plne funkčná, pripojenie k serverovej časti aj tlač dokumentov fungovali správne.

Po opätovnom vypnutí BT a reštartovaní aplikácie sa znova objavilo okno so žiadosťou o jeho zapnutie. Po odmietnutí sa aplikácia nezavrela, avšak zobrazila správu o nutnosti zapnutia BT. Aplikácia nebola funkčná a všetky tlačidla boli deaktivované. Po vypnutí aplikácie, zapnutí BT a opätovnom zapnutí aplikácie sa všetky dáta správne načítali a aplikácia bola opäť funkčná.

Záver

Cieľom práce bolo navrhnúť a implementovať riešenie pre Bluetooth tlač pomocou Android zariadení a Raspberry Pi. Pred návrhom bola nutná analýza existujúcich riešení pre Bluetooth tlač a analýza samotnej technológie Bluetoot a jej bezpečnosti.

Analýzou existujúcich riešení som zistil, že tieto riešenia nie sú vhodné pre použitie s Android zariadeniami a ich bezpečnosť nie je zaručená. Následnou analýzou technológie Bluetooth a jej bezpečnosti som zistil jej možné slabiny a bezpečnostné hrozby. Diskusiou možných opatrení proti týmto hrozbám som navrhol také opatrenia, ktoré sú vhodné pre použitie v mojej práci. S použitím protokolu Diffie-Hellman s využitím eliptických kriviek, symetrickej šifry AES a digitálnych certifikátov som vylepšil bezpečnosť technológie Bluetooth. Navrhnuté riešenie som implementoval, pričom som pre prácu so serverovou časťou vytvoril pomocné skripty pre zjednodušenie jej nasadenia a samotnej správy. Zdrojové kódy sú obsiahnuté v prílohe.

Výsledkom je serverová časť naprogramovaná v programovacom jazyku Python určená pre Raspberry Pi a užívateľská časť naprogramovaná v programovacom jazyku Kotlin určená pre Android. Celú implementáciu som zdokumentoval a na záver obsiahlym testovaním zistil jej správne fungovanie pri normálnom aj chybnom používaní.

Serverovú aj užívateľskú časť je možné v mnohých ohľadoch ešte viac vylepšiť. Pre serverovú časť by bolo prínosné implementovať komplexnejšiu správu užívateľských zariadení. Užívateľská aplikácia je taktiež veľmi jednoduchá, medzi možné vylepšenia radím napríklad možnosť zmeny hesla priamo z aplikácie alebo komplexnejšie nastavenie tlače pred odoslaním dokumentov na serverovú časť. Medzi vylepšenia užívateľskej časti je možné zaradiť aj optimalizáciu tých častí, ktoré pracujú s užívateľskými vstupmi.

Literatúra

[1] Goldtouch: *What Is A Bluetooth Dongle And Do I Need One?* [online]. [cit. 2021-04-21]. Dostupné z `https://www.goldtouch.com/usb-bluetooth-dongle-adapter/`

[2] CERVANTES, Edgar: *What is Android? Here's everything you need to know.* [online]. [cit. 2021-04-12]. Dostupné z: `https://www.androidauthority.com/what-is-android-328076/`

[3] Statcounter GlobalStats: *Mobile Operating System Market Share Worldwide* [online]. [cit. 2021-04-12]. Dostupné z: `https://gs.statcounter.com/os-market-share/mobile/worldwide`

[4] SIMS, Gary: *I want to develop Android apps — What languages should I learn?* [online]. [cit. 2021-04-12]. Dostupné z: `https://www.androidauthority.com/develop-android-apps-languages-learn-391008/`

[5] GILL, Navdeep Singh: *Kotlin vs Java: Which is Better for Android App Development?* [online]. [cit. 2021-04-12]. Dostupné z: `https://www.xenonstack.com/blog/kotlin-andriod/`

[6] ASHLEEP: *Raspberry Pi Models and Specs: A Comprehensive Guide* [online]. 2021. [cit. 2021-04-11]. Dostupné z: `https://howchoo.com/pi/raspberry-pi-models`

[7] Raspberry Pi Foundation: *FAQs* [online]. [cit. 2021-04-11]. Dostupné z: `https://www.raspberrypi.org/documentation/faqs/`

[8] Raspberry Pi Foundation: *What is a Raspberry Pi?* [online]. [cit. 2021-04-11]. Dostupné z: `https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/`

[9] Raspberry Pi Foundation: *SD cards* [online]. [cit. 2021-04-11]. Dostupné z: https://www.raspberrypi.org/documentation/installation/sd-cards.md

[10] Raspberry Pi Foundation: *Operating system images* [online]. [cit. 2021-04-11]. Dostupné z: https://www.raspberrypi.org/software/operating-systems

[11] Raspberry Pi Foundation: *Raspberry Pi Zero W* [online]. [cit. 2021-04-11]. Dostupné z: https://www.raspberrypi.org/products/raspberry-pi-zero-w/

[12] Raspberry Pi Foundation: *Raspberry Pi hardware* [online]. [cit. 2021-04-12]. Dostupné z: https://www.raspberrypi.org/documentation/hardware/raspberrypi/README.md

[13] PADGETTE, John; SCARFONE, Karen; CHEN, Lily. Guide to bluetooth security. *NIST Special Publication*, 2012, 800.121: 25.

[14] KATANGUR, Ajay K., et al. Application Level Encryption in Bluetooth. In: *Proceedings of the International Conference on Wireless Networks (ICWN)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2017. p. 3–9.

[15] TIMMINGS, Mark: *The Bluetooth device address (BD_ADDR)* [online]. [cit. 2021-04-14]. Dostupné z: https://putridparrot.com/blog/the-bluetooth-device-address-bd_addr/

[16] RAY, Brian: *Bluetooth Vs. Bluetooth Low Energy: What's The Difference?* [online]. [cit. 2021-04-14]. Dostupné z: https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy

[17] YACKEL, Ryan: *When to Use Symmetric Encryption vs. Asymmetric Encryption* [online]. [cit. 2021-04-19]. Dostupné z: https://blog.keyfactor.com/symmetric-vs-asymmetric-encryption

[18] DWORKIN, M. , BARKER, E. , NECHVATAL, J. , FOTI, J. , BASSHAM, L. , ROBACK, E. , DRAY, J. (2001). Advanced Encryption Standard (AES). *Federal Inf. Process. Stds. (NIST FIPS)*. National Institute of Standards and Technology. Gaithersburg, MD. [online]. [cit. 2021-04-19]. Dostupné z https://doi.org/10.6028/NIST.FIPS.197

[19] SULLIVAN, Nick: *A Detailed Look at RFC 8446 (a.k.a. TLS 1.3)* [online]. [cit. 2021-04-20]. Dostupné z https://blog.cloudflare.com/rfc-8446-aka-tls-1-3/

[20] CHEN, Lily, et al. *Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters*. National Institute of Standards and Technology, 2019.

[21] MERKLE, Ralph C. A certified digital signature. In: *Conference on the Theory and Application of Cryptology*. Springer, New York, NY, 1989. p. 218–238.

[22] HARN, Lein; REN, Jian. Generalized digital certificate for user authentication and key establishment for secure communications. *IEEE Transactions on Wireless Communications*, 2011, 10.7: 2372–2379.

Dodatok A

Zoznam použitých skratiek

RPi Raspberry Pi

BT Bluetooth

BLE Bluetooth Low Energy

SIG Special Interest Group

SSP Secure Simple Pairing

MITM Man In The Middle

CA Certifikačná autorita

DH Diffie-Hellman

EC Eliptická krivka

ECDH Elliptic Curve Diffie-Hellman

AES Advanced Encryption Standard

SHA Secure Hash Algorithm

TLS Transport Layer Security

IV Inicializačný vektor

CUPS Common UNIX Printing System

API Application programming interface

Dodatok **B**

Obsah priloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl` ..... zdrojové kódy implementácie
    - `server` ..... zdrojové kódy serverovej časti
    - `user` ..... zdrojové kódy užívateľskej časti
    - `cert` ..... CA použitá pri testovaní
  - `thesis` ..... zdrojová forma práce vo formáte LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce vo formáte PDF
  - `thesis.ps` ..... text práce vo formáte PS