Středoškolská technika 2016

Setkání a prezentace prací středoškolských studentů na ČVUT

Přístupový systém s Raspberry PI a Arduinem

Patrik Bílý

SPŠ A VOŠ PÍSEK
Karla Čapka 402, 397 11 Písek

Anotace

Práce se zaměřuje na problematiku zabezpečení za pomocí vývojové desky Arduino UNO, NFC a Ethernet shieldu pro tuto desku, který zajišťuje odeslání informací na server. Webový server je tvořen mikropočítačem Rasspberry Pi 2, na kterém dochází ke správě uživatelů a zobrazování příslušných informací. Například, zda-li se daný uživatel nachází v objektu, který je tímto systémem spravován nebo nikoli. Správa je řešena pomocí databáze, kterou spravuje pověřená osoba. Správce může přidělovat čipové karty konkrétním uživatelům. Data ze samotného senzoru Arduina se dostanou na server pomoci protokolu UDP a HTTP requestu. Arduino s Rasppbery Pi je připojeno kroucenou dvojlinkou do společného směrovače, přes který probíhá jejich komunikace.

Klíčová slova

Arduino, Etheren shield, NFC shield, Rasspbery Pi, Webový server, UDP

Annotation

This graduation is focused on security issues using boards Adriano, NFC Shield for the board and the Ethernet Shield, which provides sending information to the server. Web Server consists of a microcomputer Rasspberry Pi 2, which is used for managing users. Informatio about location of users in this area witch is managed. Management is solved by using a database managed by the authorized person. The administrator can add smart cards to specific users. Data from the sensor itself Arduino get to the server by using HTTP request. Arduino is conected with Raspberry by using router and network cable.

Keyword

Arduino. Etheren Shield, NFC Shield, Rasspbery, Web server, UDP

Obsah

Úvod 4

Teoretický rozbor 4

Linux 4
LAMP 4
Apache HTTP Server 4
MySQL 5
PHP 5
Raspbian 5
Raspberry Pi 6
Arduino 6
NFC 6
RFID 7
Adminer 7

Zpracování tématu 7

Praktická část 8

Instalace operačního systému 8
Instalace LAMP 8
Arduino HW 9
OBR.1- SPOJENÍ SHIELDŮ 9
Spojení Arduina s Raspberry Pi 9

Odesílání dat na server 10

Databáze uživatelů 10
Zabezpečení přenášené informace 11
Čtení čipových karet 11
Práce s oběma shieldy 12

Přijímání dat 13

Zobrazení informací 13
Program v Arduinu 14
OBR.3 – ČÁST KÓDU 15
Testování 15

Závěr 16

Použitá literatura 18

Úvod

Cílem byla komunikace mezi samotným Arduinem s jeho shieldy a webovým serverem na Rasspbery Pi, konkrétně s HTTP serverem a databází s uživateli. Konkrétní funkce celého tohoto systému je zabezpečení určitého objektu, kde můžeme vidět informace o uživatelích nacházejících se v komplexu na základě jim přidělených identifikačních karet s RFID čipem. Každému uživateli správce přidělí kartu s unikátním symbolem, následně dochází k zavedení karty do systému a přiřazení uživateli, které se provádí v databázi. Výhodou je jednoduchá správa uživatelů a malé náklady na provoz. Řešení má však i nepříznivé stránky – Rasspbery Pi se serverem musí být fyzicky spojeno za pomocí síťového kabelu s Arduinem.

Teoretický rozbor

Linux

Je operační systém založený na Linuxovém jádru. Jeho šíření je prováděno v podobě distribucí. Vzhledem k licencím, které jsou na tento software použity, se jedná o open-source, z toho tedy vyplývá, že je volně šiřitelný a je si ho možné dále upravovat i distribuovat. Tímto se liší od proprietárních uzavřených systému jako je například Microsoft Windows. Na linuxu je založen operační systém umožňující práci s Raspberry Pi, na kterém běží server. [1]

LAMP

Je softwarový balíček pro zařízení s nainstalovaným operačním systémem Linux. Existují také adaptace pro další operační systémy jako například WAMP pro Windows a MAMP pro OS X. LAMP je zkratka, která v informatice označuje sadu svobodného softwaru využívaného jako platformu pro implementaci dynamických webových stránek. Samotná zkratka znamená Linux – operační systém, Apache - webový server, MySQL – databázovým systém, PHP – skriptovací programovací jazyk. LAMP zajišťuje funkční server pro spravování databáze a práci s přijatými daty.[2]

Apache HTTP Server

Je softwarový webový server s otevřeným kódem pro Linux, OS X , Microsoft Windows a další platformy. Apache podporuje velké množství funkcí, mnoho z nich je implementováno jako kompilované moduly rozšiřující jádro. Funkce podpory programovacích jazyků na straně serveru. Podporované programovací jazyky jsou například Pearl, Python nebo PHP. Další jeho funkcí je podpora SSL, TLS a URL. Mezi podporované externí moduly pro kompresi dat webových stránek posílaných

protokolem HTTP, open source modul pro ochranu a prevenci webových aplikací před napadením. Umožnuje tvoření HTTP requestu a přijímání da z Arduina. [3]

MySQL

Je multiplatformní databázový systém. Díky své snadné implementaci lze naistalovat na Linux, Microsoft Windows, OS X. Komunikace s ním probíhá pomocí jazyka SQL. Jedná se o volně šiřitelný software, je jedním z nejvíce rozšířených databázových systémů v současné době. MySQL slouží k vytvoření databáze uživatelů na serveru.[4]

PHP

Skriptovací programovací jazyk. Je určený pro programování dynamických internetových stránek a webových aplikací ve formátech HTML, XHTML nebo WML. Využívá se také k tvorbě konzolových a desktopových aplikací. Při použití pro dynamické stránky jsou skripty prováděny na straně serveru – k uživateli je přenášen až výsledek jejich činnosti. Volání se provádí pomocí příkazového řádku, dotazovacích metod HTTP nebo pomocí webových služeb. PHP je nezávislé na platformě, rozdíly v různých operačních systémech se projevují pouze na několik systémově závislých funkcí a skripty lze mezi operačními systémy přenášet bez úprav.

Podpora mnoha knihoven pro různé účely např. Zpracování textu, grafiky, práci se soubory, přístup k většině databázových systému (MySQL, ODBC, PostgreSQL), podporuje řadu internetových protokolů (HTTP, SMTP, FTP, IMAP a další).

PHP je v současnosti nejrozšířenějším skriptovacím webovým jazykem. V kombinaci s operačním systémem Linux, databázovým systémem MySQL nebo PostgerSQL a webovým serverem Apache je často využíván k tvorbě webových aplikací. Pro takovouto kombinaci se využívá již zmíněná zkratka LAMP – tedy spojení Linux, Apache, MySQL, a PHP, Perl nebo Python. Umožňují nám pracovat se skripty, které vyhodnocují data.[5]

Raspbian

Je bezplatný operační systém založený na Debianu, který je optimalizován pro hardware Raspbrerry Pi. Operační systém obsahující základní sadu programů a nástrojů pro běh Raspberry Pi. Raspbian je dodáván s více než 35 000 balíčky s přednastaveným softwarem pro snadnou instalaci. Systém byl dokončen v roce 2012, ale stále se pracuje na zlepšení optimalizace a zvýšení výkonu. Je požívaný operační systém.[6]

Raspberry Pi

Rasspberry Pi označuje malý jednočipový počítač s deskou plošných spojů o velikosti zhruba platební karty. Primárním operačním systémem je Linux a jeho různé variace, v současné době lze používat i Windows 10 IoT Core. Deska obsahuje vývody pro monitor (HDMI), přes USB je možné připojit klávesnici a myš. V minulosti bylo vyvinuto již několik generací tohoto počítače poslední je současné Raspberry Pi 3. Modely se odlišují především výkonem, počtem konektorů a použitím. Mikroprocesor použitý na desce je z rodiny ARM, takže je srovnatelný s běžným smartphonem. Na rozdíl od desky Arduino je možné Raspberry Pi použít nejen k ovládání různých zařízení pomocí GPIO kontaktů, ale i k samotnému vývoji aplikací. Lze ho také použít jako multimediální centrum pro distribuci videa a hudby nebo jen pro přístup k internetu. Samotný mikropočítač tvoří server. [7]

Arduino

Arduino je označení pro malé jednodeskové počítače založené na mikrokontrolerech ATmega od firmy Atmel. Nejedná se tedy o počítač ve smyslu stolního počítače nebo chytrého telefonu. Nelze k němu proto snadno připojit monitor ani klávesnici či myš, ale je připraven na připojení LED diod, displeje z tekutých krystalů, servomotorů atd. Jedná se o otevřenou platformu s vývojovým prostředím, která vychází z prostředí Wiring a Processing, prostředí pro výuku programování. Na rozdíl od Raspberry Pi není Arduino zamýšleno jako plnohodnotný stolní počítač. Řídící program je vyvíjen zvlášť na počítači a do Arduina je poté nahrán a spuštěn. Díky tomuto má nízkou spotřebu a hodí se například pro řízení robotů. Každá deska má většinu I/O pinů přístupných přes standartní patice, do kterých se jednoduše připojují další obvody, kterým se říká shieldy. Na deskách bývá několik diod, restartovací tlačítko, konektor pro ISCP programování, napájecí konektor, oscilátor a obvody zprostředkovávající komunikaci po USB. Je určené pro čtení a odesílání dat na server.[8]

NFC

NFC (Near fiedl comunication) je modulární technologie radiové bezdrátové komunikace mezi elektronickými zařízeními na velmi krátkou vzdálenost. Tato komunikace je založena na standardech RFID zahrnující ISO/IEC 14443. Standardy pro tuto komunikaci byly definovány neziskovou organizací NFC Forum v roce 2004. Technologie umožňuje obou cestnou komunikaci mezi koncovými zařízeními. Systémy používané v minulosti byly založeny na čtení bezkontaktního čipu RFID a umožňovaly pouze jednocestnou komunikaci. NFC Forum má zájem o to, aby se potenciální NFC zařízení používala jako elektronické identifikační karty a klíčenky, dosud používané pouze pro kontrolu vstupu. Mohou být použity pro čtení NFC shieldem. [9]

RFID

RFID (Radio Frequency Identification, Identifakace na rádiové frekvenci) je další generace identifikátorů navržených k identifikaci. Čipy jsou k dispozici v provedení pro čtení nebo pro čtení a zápis. Komunikace funguje na základě využití nosné frekvence 125 kHZ, 134 kHz nebo 13,56 MHz. Technologie RFID využívá a vylepšuje novější sytém NFC, rozšiřuje jeho možnosti. RFID čipy obsahují unikátní 96bitové číslo takzvané EPC. Vzhledem k délce 96 bitů může EPC nabídnout dostatečný číselný prostor. Použité karty v práci jsou právě toho typu.[10]

Adminer

Je nástroj vytvořený v jazyce PHP umožňující prostřednictvím webového rozhraní jednoduchou správu databáze MySQL. Adminer je alternativa k phpMyAdmin. Oproti phpAdminu je při běžných operacích zhruba 2,5 rychlejší, při vzdáleném spojení je rozdíl rychlosti ještě vyšší. Obsahuje 31 kompletně přeložených jazykových verzí včetně Češtiny. Slouží jako rozhraní pro správu databáze.[11]

Zpracování tématu

Instalace operačního systému Raspbian a LAMP serveru na Rasperry Pi je základním krokem pro vytvoření příjímací strany pro data odeslaná z Arduina. Důležitou součástí je také zabezpečení přenosu

dat z Arduina na server. O správu uživatelů se stará databáze typu MySQL spuštěná na serveru. Jedině administrátor může upravovat databázi, spravovat uživatele. Informace, které jsou vyhodnoceny a zobrazují na stránce *info.php* si může zobrazit pouze osoba, která zná přístupové heslo. Přenos je zabezpečen tak aby nedošlo k přihlášení uživatele v době, kdy nepřiloží svou identifikační kartu. Uživatel je o správném přenesení dat informován stavovými diodami, které dále indikují, jestli je daná čipová karta uložená v databázi.

Praktická část

Instalace operačního systému

Prvním krokem je instalace operačního systému RASPBIAN JESSIE na Rasspberry Pi. Pomocí programu *RPi-sd card builder v1.2* na paměťovou kartu se nahraje instalační soubor ve formátu .img a program sám vytvoří bootovací kartu, která je po vložení do slotu pro micro SD karty na spodní straně samotného mikropočítače připravena na instalaci OS. Po připojení Rasspebrry Pi na napájení sama naistaluje systém a nastartuje do grafického prostředí. Dojde k vytvoření administračního účtu a výchozímu nastavení systému.

Administrátorský účet není po spuštění chráněn heslem, tudíž je prvním krokem po nastartování systému nastavení hesla pro přihlášení. Systém může běžet v grafickém nebo v textovém režimu, vzhledem k tomu, že bude možné i na samotném Raspberry Pi zobrazit informace z webového serveru byl zvolen grafický režim i pro větší přehlednost v systému. V sytému si pomocí příkazů *sudo adduser patrik* vytvoříme uživatelský účet a nastavíme si také přihlašovací heslo. Je třeba také přidat uživateli práva a to za pomocí *sudo nano /etc/sudoers*, kde připíšeme pod uživatele pi *patrik ALL = (ALL) NOPASSWD: ALL.* Uživatel patrik by měl mít poté všechna práva. Toto je základní nastavení, které budeme potřebovat pro funkčnost celé práce. Můžeme také nastavit statickou adresu pro Raspberry, ale není to nutné, pokud poběží celá sestava stále na stejném směrovači, kde je nastaveno přidělené IP adresy na vždy.

Instalace LAMP

Samotná instalace se provádí přes Terminál. Nejdříve se pomocí příkazu *sudo aptitude install apache2* naistaluje se a spustí Apache, který má v sobě již nadefinovaný soubor index.html, aby bylo možné ověřit jeho funkčnost. Příkaz *sudo* na začátku říká systému, že má příkaz spustit s "rootovskými" oprávněními, příkaz *aptitude* spouští aplikaci se stejným názvem, která se stará o správu balíčků, za ním následuje příkaz *install,* který říká, že budeme instalovat novou aplikaci a na konci následuje jméno balíčku, tedy *apache2.* Pomocí příkazu zjistíme IP adresu Rasspberry Pi v síti a vložíme ji do adresního řádku v prohlížeči a poté se nám otevře již zmiňovaný soubor index.html. Při instalaci se vytvoří složka,

do které budeme moci vložit svojí HTML stánku. Dalším krokem je instalace MySQL databázového systému za pomocí příkazu *sudo aptitude install mysql-server php5-mysql*. Instalace probíhá za pomocí dvou balíčků, první je samotná databáze, druhý balíček je PHP modul, který dokáže s databází komunikovat. Pro správu databáze jsem zvolil rozhraní Adminer. Poslední částí je doinstalování PHP do Apache, abychom mohli používat skripty pro přenos dat a práci s informacemi z Arduina a pro komunikaci s databází. Pro instalaci PHP je použito *sudo aptitude install libapache2-mod-php5 php5*. Jako při předchozích instalacích se jedná opět o dva balíčky, jsou jimi balíček *php5*, tedy samotný interpret a balíček *libapache2-mod-php5,* který zajišťuje propojení interpreta se serverem. Vzhledem k napojení Raspberry Pi do sítě pomocí síťového kabelu přes směrovač a faktu, že je spuštěn LAMP, je možné se na server připojit i z jiného počítače umístěného v síti. Samotné informace zobrazované na serveru mohou být tedy zobrazeny na monitor dvěma způsoby, buď na monitoru připojeném k Raspberry Pi za pomocí HDMI kabelu nebo na jiném počítači v síti, který bude znát IP adresu serveru.

Arduino HW

Na Arduinu je připojen NFC shied a na něj dále Ethernet shield. Do Arduina nejsou připojeny všechny piny z NFC shieldu, protože oba shieldy v základní konfiguraci využívají stejný pin a docházelo by ke kolizím. Tudíž bychom nemohli používat oba shieldy spolu. Z NFC shiedu není do Arduina připojen devátý pin (viz Obr. 1), ten je nahrazen pinem deset. Z Etherent shieldu není do NFC shieldu připojen desátý pin, ale je za pomocí kablíku přiveden místo desátého pinu NFC do Arduina. Dále musí také dojít k softwarové úpravě a předefinování těchto pinů v samotném programu.

OBR.1- SPOJENÍ SHIELDŮ

Spojení Arduina s Raspberry Pi

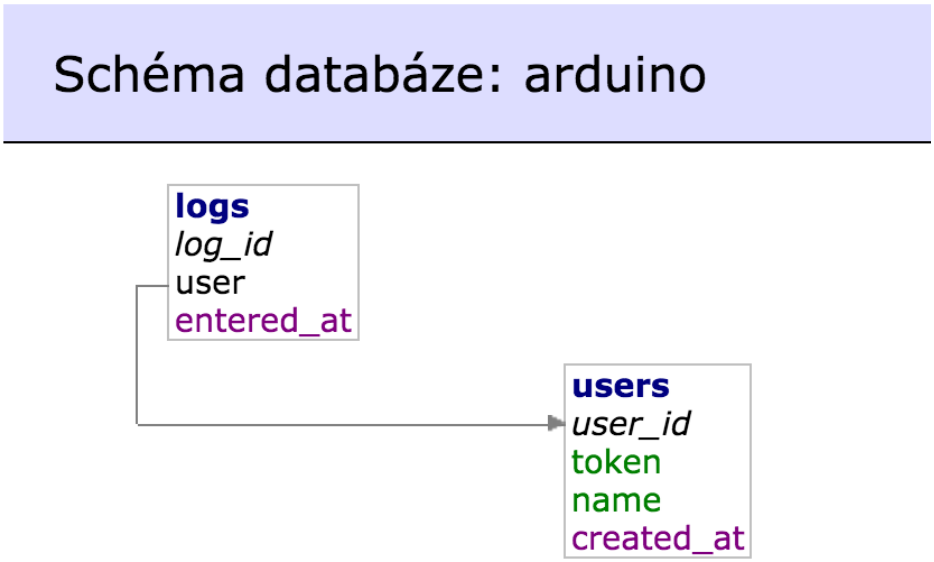
Na Arduinu je připojen NFC shied a na něj dále Ethernet shield. Do Arduina nejsou připojeny všechny piny z NFC shieldu, protože oba shieldy v základní konfiguraci využívají stejný pin a docházelo by ke kolizím. Tudíž bychom nemohli používat oba shieldy spolu. Z NFC shiedu není do Arduina připojen devátý pin, ten je nahrazen pinem deset. Z Etherent shieldu není do NFC shieldu připojen desátý pin, ale je za pomocí kablíku přiveden místo desátého pinu NFC do Arduina. Dále musí také dojít k softwarové úpravě a předefinování těchto pinů v samotném programu.

Odesílání dat na server

Po přiložení čipové karty se vezme její číslo a dále se používá jako *cardID.* V programu je pevně definovaný klíč, který zná jen Arduino a server. Na server se odesílá číslo karty, tedy *cardID*, a také *hash* vytvořený za pomocí algoritmu MD5, který obsahuje klíč, číslo karty, a náhodnou hodnotu (čas). Takto vytvořený *hash* se spolu s číslem karty odešle za pomocí *HTTP requestu* na server, kde se s ním dále pracuje. Data se vysílají jako *string*, který obsahuje *cardID* a *hash* vytvořený při odeslání. Po přijetí dat se na serveru ověří, jestli byla skutečně přiložena karta nebo někdo zachytil odeslaný paket a znovu ho použil. Do Arduina se bude vracet poté ze serveru *response*, která bude závislá na vyhodnocení porovnávání kontrolního *hashe* a přijatého *hashe* z Arduina. Pokud budou stejné odešle se zpět do Arduina *true* a rozsvítí se zelená indikační dioda, s daty se bude dále pracovat. Při neshodě při porovnávání se do Arduina odešle *false*, to znamená, že oba *hashe* nejsou stejné a rozsvítí se červená indikační dioda.

Databáze uživatelů

Správa uživatelů je řešena za pomocí databáze, konkrétně MySQL. V databázi jsou vytvořeny dvě tabulky (viz Obr.2) , které jsou navzájem provázány. První tabulka se týká uživatelů, kde probíhá jejich správa, přidání, odebrání a jejich úpravy. Každému uživateli je přiděleno *user_id*, čas, kdy byl vytvořen, jméno a číslo čipové karty. Druhá tabulka se stará o informace, které se odešlou na server, vždy když uživatel přiloží svojí identifikační kartu, která mu byla přidělena. Přidání probíhá tím způsobem, že se odešla na server *cardID*, které se následně vloží do databáze a přiřadí se k němu uživatel. Po přijetí dat z Arduina databáze na základě informací o čipové kartě vyhodnotí jaké *user_id* je s touto kartou spojené a zobrazí jméno uživatele, datum a čas přiložení karty. Vzhledem k faktu, že každá čipová karta má své unikátní identifikační číslo, nemůže nastat záměna uživatelů při načtení karty na NFC čtečce. Po načtení každé karty je do Arduina odeslán response *true* nebo *false*. Hodnota závisí na existenci čipové karty v databázi a také na správném přijetí dat na server. Když nebude kontrolní *hash* vytvořený na serveru souhlasit s přijatým *hashem* s daty, nebude se dále pracovat a do Arduina se vrátí f*alse* . Pokud se karta v databázi nenachází, odešle se *false,* jestliže se daná čipová karta v databázi nachází, odešle na zpět do Arduina response *true* a pokračujeme dále k přiřazení karty k uživateli, který je vytvořen v databázi a má přidělené číslo načtené čipové karty.



OBR.2-.SCHÉMA DATABÁZE

Zabezpečení přenášené informace

Pro větší bezpečnost a vyloučení možnosti, že osoba snažící se vstoupit pod čipovou kartou, která mu nepatří a odchytnul by paket odeslaný na server. Na Raspberry je nainstalovaný NTP server, který se synchronizuje s Arduinem. Do programu je následně uložen klíč (*key*), který zná jen Arduino a Raspberry. V Arduinu se vytvoří md5 *hash*, který se skládá z (*key, cardI, time).* Když se data dostanou na server a zpracují se, vezme se *cardID,* které přišlo z Arduina, a na serveru se vytvoří porovnávací hash skládající se z (*key, cardId , time*). Časy na obou zařízeních jsou synchronizované. *Hash* vytvořený na serveru se porovná s *hashem* přijatým z Arduina. V případě, že jsou stejné, server odešle do Arduina na zpět response *true*, tedy vše je v pořádku. V případě, kdy dojde k neshodě při porovnávání, odešle response f*alse*. Dochází tedy aspoň k základnímu zabezpečení proti útočníkům, kteří by mohli odposlouchávat síťový provoz a odchytávat pakety. Při opětovném odeslání paketu by se lišil porovnávací *hash* vytvořený na serveru, protože paket, který by útočník odchytl, byl vytvořen již dříve a byl použit při *hashování* jiný čas než při vytvoření kontrolního *hashe* na serveru po přijetí odchyceného paketu, který odeslal útočník.

Čtení čipových karet

Program pro čtení. Po každém načtení karty a odeslání na server dojde ke kontrole správnosti dat, která je zmíněna v časti zabývající se odesíláním na server. Pokud vše proběhlo v pořádku, vyhodnotí se, jestli je daná karta uložena v databázi, když ano, odesílá zpět do Arduina *respons true,* pokud není

karta v databázi, odesílá se *false*. Pokud Arduino přijme response *true*, nastaví pin, na kterém je připojena led dioda, na log 1 a tím rozsvítí diodu připojenou na tento pin, pokud se do vrátí *false,* dojde také k vyhodnocení, ale rozsvítí se jiná indikační dioda a na serveru se objeví informace, že daná karta není v databázi přiřazena žádnému uživateli. Správce je tedy informován hláškou a uživatel s kartou nepřidělenou v databázi bude upozorněn led diodou. Může nastat situace, kdy přijde někdo jiný s čipovou kartou, která mu nepatří, ale je přidělená jiné osobě. Tento problém způsobuje přenositelnost a nepevné vázaní karty k uživateli. Karta slouží jen k informaci o přihlášení osoby do systému na základě a pouze přiložení čipu s číslem, nemůže určit, jestli se jedná právě o majitele karty nebo osobu, která kartu odcizila a používá jí bez vědomí majitele. Jediná možnost, jak se tedy do systému přihlásit, je již zmiňované odcizení čipové karty majiteli. V případě, že útočník odchytí odeslaný paket s daty z Arduina, je zavedeno bezpečnostní opatření, které již bylo zmíněno v předchozím textu.

Práce s oběma shieldy

Oba shieldy v základním nastavení používají desátý pin pro komunikaci, muselo dojít k již zmiňované hardwarové úpravě. Nezbytná je také softwarová úprava v samotném programu. Do knihoven, které jsou pro tyto shieldy určeny nebyl nutný žádný zásah. V programu bylo třeba nadefinovat devátý pin u NFC shiedlu příkazem pin #define PN532_SS 9. U Ethernet shieldu se musel nadefinovat desátý pomocí příkazu pin #define ETH_SS 10. Toto bylo jen základní předefinování pinů, aby mohla data až do samotného Arduina. Oba dva shieldy nemohou být používány najednou, musí se mezi nimi přepínat. Vždy shield, s kterým nechceme v danou chvíli komunikovat, odpojíme od SPI sběrnice, aby se mohla předávat data mezi druhým shieldem a základní deskou. Celé přepínaní je realizováno pomocí těchto příkazů *pinMode(PN532_SS,OUTPUT); pinMode(ETH_SS,OUTPUT);* **digitalWrite(PN532_SS,HIGH);** *digitalWrite(ETH_SS,HIGH);*. Tím dojde k odpojení shiedlu od sběrnice SPI. Musí také dojít k přenastavení SPI, protože každý shield využívá jiné. NFC shield využívá *LBSFIRST* a Etherent *MBSFIRST*. Dále se musí změnit i nastavení hodinového děliče, které je pro shiedly rozdílné, pro NFC je to *SPI_CLOCK_DIV64* a pro Etherent *SPI_CLOCK_DIV4*. Přepínání se musí využívat vždy, když chceme používat jiný shield než ten, s kterým zrovna pracujeme. Přepínání se dále poté řeší jen voláním této části programu vždy před blokem, kdy chceme pracovat s jiným shieldem. Tohle je jedna z možností, jak používat více shieldů na SPI sběrnici, které v základu využívají pro přenos dat stejný pin. Další možností je použít shiedly, které nevyužívají stejný pin nebo upravené knihovny a za pomocí kablíků přepojit SPI sběrnici na digitální piny a ty také definovat v kódu.

Přijímání dat

Po připojení Arduina do sítě a zapojení napájení se spustí nahraný program, který se připojí na server. Po úspěšném připojení se může přiložit čipová karta. Program v Arduinu se jako první připojí na server, po úspěšném připojení načte čipovou kartu. Na server se za pomoci HTTP requestu a php skriptu, který vezme informace metodou GET z URL, kde budou za sebou vypsána data, nejdříve bude proměnná, které se bude týkat data umístěných v URL za ní. Dostaneme číslo karty (cardID), *hash* a time. Hash na serveru je vytvořen z předem definovaného klíče, cardID a času odeslání, který je synchronizován se serverem. Čas by zvolen vzhledem k tomu, aby byla hodnota vstupující do hashovacího algoritmu spolu s cardID a klíčem vždy v každém okamžiku opravdu náhodná. Po přijetí těchto informací se na serveru z přijatého cardID vytvoří pomocí hashovacího algoritmu md5 kontrolní *hash*, který se vytvoří z klíče, cardID a času. Klíč zná jen Arduino a Raspberry, z hashe se nedá získat. Po porovnání přijatého a kontrolního *hashe* se podle výsledku odešle response zpět do Arduina, se kterým se bude dále pracovat. Když budou oba *hashe* stejné, odešle se zpět *true* a rozsvítí se zelená led dioda, pokud se budou *hashe* lišit, do arduina se vrátí *false* a rozsvítí se červená led dioda. Pokud dojde ke správnému přijetí, databáze si vezme cardID a to přiřadí konkrétnímu uživateli, který již v databázi existuje. Po přiřazení dojde k zobrazení informace na stránce *info.php*, kde jsou vidět všechny záznamy správně přijatých karet. Administrátor a osoba, která zná heslo potřebné pro vstup na stránku *info.php,* bude moci vidět záznamy.

Zobrazení informací

Informace jsou zobrazeny na samostatné stránce *info.php*, do které je přístup zabezpečen pomocí hesla. Dochází k přenosu informací mezi touto stránkou a databází. Po zadání hesla můžeme vidět jména uživatelů a čas, kdy byla karta načtena. První informace je jméno uživatele následuje datum a přesný čas i se sekundami. Zobrazuje se nám pouze informace o tom, kdy se čipová karta načetla. V databázi může dále administrátor vidět, kdy byla karta přidána do systému a přidělena k uživateli. Nezískáváme informace o odchodu uživatele, pouze o jeho příchodu. Zobrazovaní informací je volně dostupné na síti po připojení na server otevření php *scriptu info.php* a přihlášení se pod heslem. Informace o přihlášených uživatelích může tedy vidět pouze administrátor a pověřená osoba. Informace by mohli být také umístěny na monitoru u čtečky karet, kde by byla otevřena pouze tato stránka, aby měla osoba, která přiložila čipovou kartu, informace, jestli byla skutečně přihlášena do systému.

Program v Arduinu

Program byl vytvořen ve vývojovém prostředí Arduino 1.6.8, které je doporučeným vývojovým a nahrávacím prostředím pro práci s deskami Arduino. Prvním krokem při tvorbě je přidání knihoven do programu. Použité knihovny slouží pro práci se shieldy a pro další použité funkce, jako například pro vytvoření NTP žádosti na synchronizování času se serverem. Použité knihovny, které jsou vloženy do programu nebyly nijak upravovány. Většina knihoven je již součástí Arduina nebo se pomocí pár kliknutí přidají přímo ve vývojovém prostředí Ardruina. Knihovna pro NFC shield se musí stáhnout na stránkách výrobce a vložit soubor s příponou zip do složky Arduino/libreries, knihovna bude po vložení dostupná ve vývojovém prostředí u ostatních knihoven. Další přidané knihovny byly staženy ze serveru GitHub od autora Links2004 a jako u předchozí knihovny vloženy do složky a přidány do programu. Po přidání všech knihoven je dalším krokem definování již zmíněných pinů, které musí být upraveny, aby mohli oba shieldy spolu fungovat. Musíme také nastavit, aby se použité piny změnily v celém programu pomocí *PN532 nfc(PN532_SS),* který všude nahradí devátý pin místo desátého. Dalším krokem je vytvoření ethernet clienta a nastavení *mac* adresy. Mac adresa shieldu může být jakákoli, protože v síti nepoužíváme žádný další Etherent shield, jinak by se musely adresy lišit. Přidáme klíč, který se používá při odesílání dat a při vytváření hashe. Nezbytným krokem je nastavení IP adresy serveru, se který budeme komunikovat. Posledním krokem před *setupem* je nastavení UDP. V setupu nastavíme rychlost sériové komunikace a hned poté přepneme shield a přidáme *delay*, aby měl Ethernet shiedl čas na spuštění. Po naběhnutí vytiskneme IP adresu přidělenou Ethernet shieldu a přepneme na NFC shield. Následně po přepnutí na NFC nejdříve provedeme kontrolu, jestli je shield připojen. Pokud je shield připojen, pokračujeme dále do další části, kde čekáme na odpověď ze serveru a musíme tedy přepnout na Ethernet shield. Pokud se dočkáme odpovědi, tak ji vytiskneme do *serial* monitoru, pokud odpověď nepřijde, pokračujeme tím, že přepneme na NFC shield a začneme číst přiloženou čipovou kartu. Po načtení je číslo karty používáno v programu jako cardID. Dalším krokem je vygenerování času, který budeme dále používat. To tím způsobem, že odešleme žádost o synchronizaci s NTP serverem. Přepneme na Ethernet shield a vytvoříme *hash* za pomocí hashovacího algoritmu md5. Po vytvoření hashe již následuje samotné odeslání dat na server pomocí HTTP requestu. Pokud se připojení nepovede, vytiskneme do *serial* monitoru *connection to server failed,* jinak se pokračuje dále. V dalším bloku kódu máme dříve zmiňované přepínání mezi shieldy, které se využívá v celém programu. Pokračujeme na část, která se zabývá NTP clientem, který nám obstarává synchronizování času s NTP serverem spuštěném na Raspberry Pi. Poslední část programu se věnuje vytváření UDP paketu pro odeslání dat na server. (viz Obr. 3)

```
if (cardId != 0) {
  digitalWrite(ledWorkingPin,HIGH);
  Serial.print("Read card #");
  Serial.println(cardId);

  adjustTime(getNtpTime());

  spiSelect(ETH_SS);
  ethClient.stop();
  if (ethClient.connect(server, 80)) {

    time_t hastTime = now();

    String hashInput = key + cardId + hastTime;
     char hashInputArr[45];
    hashInput.toCharArray(hashInputArr,45);
    unsigned char* hash=MD5::make_hash(hashInputArr);
    char *md5str = MD5::make_digest(hash, 16);

    ethClient.print("GET /dlouhodobka.php?cardId=" ); ethClient.print(cardId); ethClient.print("&hash=" ); ethClient.print(md5str); ethClient.print("&inputhash=" );
    ethClient.println("HTTP/1.1");
    ethClient.println("Host: ");
    ethClient.println("User-Agent: arduino-ethernet");
    ethClient.println("Connection: close");
    ethClient.println();
    free(md5str);
```

OBR.3 – ČÁST KÓDU

Testování

Testováním proběhne ověření funkčnosti výsledného výrobku v podobě načtení čipové karty, odeslání na server, zpracováním databází a zobrazování dat.

Arduino:

- Připojení na server funguje
- Modul čte čipové karty
- Spojení obou shieldů funguje a lze je používat dohromady
- Program vytvoří hash pro bezpečný přenos
- Odesílání dat na server pomocí HTTP requestu probíhá v pořádku
- Indikační diody pracují správně podle programu
- Do Arduina se vrací zpět respons a dál se s ním pracuje

Raspberry Pi:

- Operační systém funguje
- LAMP server je funkční
- Server je dostupný ze sítě
- Databáze funguje a vyhodnocuje data přijatá z Arduina
- Systém má uživatelské rozraní, které zajišťuje Adminer
- Lze spravovat uživatele
- Pověřená osoba má přehled o přihlášených osobám

Závěr

Výsledkem dlouhodobé maturitní práce je funkční sestava, která slouží k evidenci a identifikaci uživatelů za pomocí čipové karty. Jako server, na kterém dochází ke správě uživatelské databáze, je použito Raspberry Pi s nainstalovaným LAMP serverem. Čtení čipových karet se realizuje za pomocí spojení Arduina, NFC shieldu a Etherent shieldu, kde Ardino slouží jako výpočetní prvek, NFC shield čte přiložené čipové karty a Etherent shield zajišťuje připojení do sítě. K propojení těchto dvou samostatných prvků slouží směrovač, který zabezpečuje i přístup jiných uživatelů na server v síti. Při spojování NFC shiedlu s Ethernet shieldem došlo k problému v důsledku použití stejného pinu pro přenos informací mezi více deskami při základním natavení. Muselo dojít tedy k hardwarové i softwarové modifikaci obou modulů. Program nahraný do Arduina v sobě kombinuje čtení čipových karet a odesílání informací na server za pomocí HTTP *requestu*. Po odeslání dat na server se vrací do Arduina response, který nás informuje o správném přenesení dat a přijetí na serveru. K zobrazení odpovědi ze serveru slouží dvě notifikační diody, zelená nám říká, že data byla přijata v pořádku, červená dioda indikuje nesprávný přenos dat na server. Pro informování o probíhajících procesech v Arduinu je použita žlutá notifikační dioda. Pro zabezpečení přenosu se na server odesílá číslo čipové karty a vytvořený *hash* za pomocí algoritmu md5. *Hash* z Arduina obsahuje klíč, který je definován v programu, číslo čipové karty a čas odeslání. Čas je synchronizován s NTP serverem. Pro ověření se na serveru vytvoří kontrolní *hash* z klíče, přijatého čísla karty a času. Ověření se provádí porovnáním přijatého a vytvořeného kontrolního hashe na serveru, když dojde ke shodě pokračuje se dál a přiřadí se číslo karty uživateli a na Arduinu se rozsvítí zelená notifikační dioda. Pokud není shoda mezi *hashy* s informacemi se dále nepracuje. Pro správu uživatelů je na serveru vytvořena MySQL databáze, která se skládá ze dvou tabulek a je spravována pomocí webového rozhraní Adminer. Správou databáze je pověřen administrátor, který zná přihlašovací údaje a může tedy celou databázi editovat. Informace se zobrazují na samostatné stránce *info.php*, na kterou je zabezpečen vstup pod heslem, kde jsou zobrazeny jména uživatelů a časy, kdy se do systému přihlásili.

Identifikace uživatele je sporná, protože se k ní využívá pouze jeden údaj. Čipová karta, kterou má každý uživatel přidělenou a provázanou s jeho jménem v databázi je přenositelná, může dojít k jejímu odcizení a použití bez vědomí majitele. Nemůžeme tedy jen podle přiložené karty a odeslání její identifikační hodnoty na server určit, jestli se skutečně jedná o majitele. Vždy ale musí být přiložena čipová karta, kdyby osoba snažící se přihlásit do systému odchytla odesílaný paket z Arduina, do systému se nedostane, protože proti tomuto typu útoku jsme chráněni odesílám *hashe,* který je vytvořen pomocí hashovacího algoritmu md5 z klíče (key), carID(čísla karty) a time (časového údaje). Ověření na serveru spočívá ve vytvoření kontrolního *hashe*, který se s přijatým porovná. Pokud by

útočník odchytl odeslaný paket, je v něm použit jiný time při hashovaní a tím se hodnota hashe liší a při porovnání by nedošlo ke shodě.

Sestava se dá použít i jako docházkový systém, kdy nám bude ukazovat, jestli se daná osoba přihlásila a v jaký čas. Nebo jen jako identifikace při vstupu do určitého místa. Databáze se dá stále rozvíjet, vylepšovat a upravovat podle konkrétních požadavků na funkčnost.

Použitá literatura

[1] Linux. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/Linux

[2] LAMP. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/LAMP_(software_bundle)

[3] Apache HTTP Server. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/Apache_HTTP_Server

[**4**] MySQL. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/MySQL

[5] PHP. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/PHP

[6]Raspbian.Raspbian[online]. [cit. 2016-03-29]. Dostupné z: https://www.raspbian.org/RaspbianFAQ

[7] Raspberry Pi. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/Raspberry_Pi

[8] Arduino. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/Arduino

[**9]** Near field communication. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation,2016[cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/Near_field_communication

[10] Radio-frequency identification. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/Radio-frequency_identification

[11] Adminer. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://en.wikipedia.org/wiki/Adminer